ZSM Security: Threat Surface and Best Practices

Chafika Benzaïd* and Tarik Taleb†
Aalto University, Finland
Email: *chafika.benzaid@aalto.fi, †tarik.taleb@aalto.fi

***Abstract*—The ETSI's Zero touch network and Service Management (ZSM) framework is a prominent initiative to tame the envisioned complexity in operating and managing 5G and beyond networks. To this end, ZSM framework promotes the shift towards full Automation of Network and Service Management and Operation (ANSMO) by leveraging the flexibility of SDN/NFV technologies along with Artificial Intelligence, combined to the portability and reusability of model-driven, open interfaces. Besides its benefits, each leveraged enabler will bring its own security threats, which should be carefully tackled to make ANSMO vision a reality. This paper introduces the potential ZSM's attack surface and recommends possible mitigation measures along with some research directions to safeguard ZSM system security.**

***Index Terms*—5G, ZSM, AI, ML, Security, SDN, NFV**

I. Introduction

The upcoming fifth generation of mobile communication networks (5G) are envisaged to notably improve network capabilities by providing ultra-low latency, ultra-high bandwidth, ultra-reliability, seamless coverage and greater connection density. 5G capabilities will pave the way for new use cases (e.g., virtual reality, autonomous vehicles, etc.), unlocking new business models. To meet the diverse and varying requirements of the foreseen use cases, 5G networks are being designed as highly programmable, extremely flexible and holistically-managed infrastructures that are service- and context-aware. For this purpose, emerging enablers such as Software Defined Networking (SDN), Network Function Virtualization (NFV) and Multi-access Edge Computing (MEC) are identified as the foundational pillars for designing 5G network architecture. However, the increased level of flexibility, programmability and efficiency brought by these enablers will come at the price of higher complexity in operating and managing 5G networks. To tame this complexity, a shift towards full Automation of Network and Service Management and Operation (ANSMO) is imperative. One prominent initiative to make the vision of full automation a reality is undertaken by the ETSI's Zero touch network and Service Management Industry Specification Group (ZSM ISG). The main goal of the ETSI ZSM ISG is to specify a reference architecture that supports zero-touch end-to-end smart network and service management in next-generation networks. The ZSM framework will leverage SDN/NFV technologies and Artificial Intelligence (AI) and Machine Learning (ML) techniques to empower self-managing functionalities (e.g., self-configuration, self-healing, self-optimization and self-protecting). To meet this goal, a set of architectural principles is directing the design of the ZSM framework, that are being service-based, policy-driven, modular, extensible, scalable, and resilient to failures [1].

The full ANSMO envisaged by ZSM will give rise to several benefits, including lower OPerating EXpenses (OPEX), accelerated time-to-value, and reduced risk of human error. Meanwhile, a major challenge facing ANSMO is to protect networks, services and data against theft and attacks. Indeed, the risk of full automation is the ability to broadly and rapidly replicate a small isolated error, putting the entire ecosystem into peril. The ANSMO attack surface is broad as it relies on several technologies and concepts (e.g., virtualization, programmability, automation, AI/ML); each of them bringing its own security threats, which need to be carefully addressed.

This paper presents the potential security threats that may hinder a ZSM system and the best practices to cope with them. The rest of the paper is organized as follows. Section II briefly describes the ZSM reference architecture. Section III discusses the main security risks associated with the different ZSM's key enabling technologies, and illustrates some possible attack scenarios. Section IV recommends a set of best practices to safeguard ZSM system security, while highlighting some research directions. Finally, the paper concludes in Section VI.

II. ZSM Architecture

The ZSM framework's reference The ZSM framework's reference architecture [1] is devised to empower full automated network and service management in multi-domain environments that include operations across legal operational boundaries. As depicted in Fig. 6, the framework architecture integrates multiple management domains (MDs), an End-to-End service MD, intra- and cross-domain integration fabrics and cross-domain data services.

Each MD is in charge of smart automated management of resources and services within its scope. The End-to-End service MD deals with the management of end-to-end services across multiple administrative domains. The separation between MDs and the End-to-End service MD fosters system modularity and enables their independent evolvement. Each MD, including the E2E service MD, consists of several management functions grouped into logical groups (e.g., domain collection services, domain intelligence services, domain analytics services, domain control services and domain orchestration services) and exposes a set of management services via service interfaces. Services local to MD are provided and consumed inside the domain using the intra-domain integration fabric. Meanwhile, services exposed cross-domains are consumed through cross-domain integration fabric. Data in Cross-domain Data Services can be leveraged by intelligence services within MDs and End-to-End service MD to empower domain-level and cross-domain AI-based closed-loop automation, respectively.

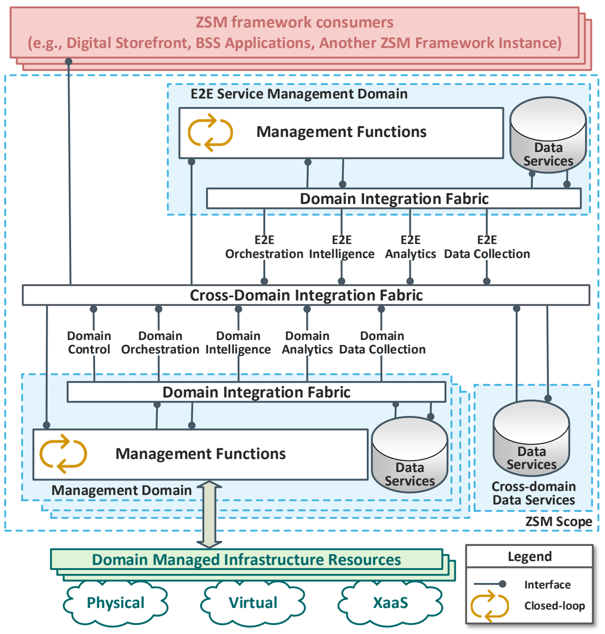


Fig. 1: The ZSM Reference Architecture [1].

III. ZSM Attack Surface

In this section, we discuss the main security threats stemming from the different ZSM's key enabling technologies and illustrate some possible attack scenarios.

A. Open APIs' Security Threats

Application Programming Interfaces (APIs) are an emerging technology for integrating applications using web technology. APIs are a key enabler for ZSM framework allowing communication and interfacing between framework's components and services. ETSI ZSM ISG recommends Open APIs as an architecture principle of the ZSM framework [1].

APIs play an integral part during provisioning, management, orchestration and monitoring of the services running in the ZSM framework, which makes them a perfect target for attackers. According to Gartner [2], API abuses will be the most-frequent attack vector by 2022. An attacker may try to exploit insecure APIs for accessing or tampering ZSM's services and/or databases. The attacker can be a compromised MD, a compromised E2E service MD or a digital storefront malicious customer. APIs-based attacks can lead to data loss/leakage, identity theft, system compromise, as well as service unavailability. Potential API-based attacks against ZSM framework include *parameter attacks*, *identity attacks*, *Man-In-The-Middle attacks* and *(Distributed) Denial of Service attacks*.

1) Parameter attacks: Parameter attacks exploit the data sent into an API, including URL, query parameters, HTTP headers and/or post content. An example of such attacks is the *injection attacks* (e.g., SQL, NoSQL, JSON, code). Improperly validated parameters within the URL, header and body of the APIs may lead to injection attacks.

One potential target of injection attacks in the ZSM framework is the Common Data Services component. This latter holds management data (e.g., performance monitoring data, assurance data, configuration data, and network/service topology data) received from the Management Domains (MDs) including the E2E service MD in the ZSM framework. Data in the Common Data Services are used to different aspects of automated E2E optimization as well as automation of management and operation of the whole ZSM framework [1]. The prominent role played by the Common Data Services in automation process makes it a valuable target for attackers. In fact, the exposed interface to query the "Common Data Services" can be source of injection attack. Indeed, if a malicious query is passed directly into the "Common Data Services" without input sanitization, the data will be subject to different security threats; such as unauthorized data access, data manipulation (i.e., insertion, modification and deletion), and Denial of Service (DoS). For instance, an attacker can manipulate the network/service topology data to insert fake links and/or malicious nodes or services in the topology. If we assume that the attacker is a malicious MD, the injection attack may allow him to access the detailed data of another MD. A DoS attack could be triggered by injecting a script in the malicious query, whose execution can make the Common Data Services unresponsive. Note that the aforementioned threats are also valid for Data Services of a given MD.

2) Identity attacks: Identity attacks exploit flaws in authentication, authorization and session tracking. Many applications publishing APIs require clients to use an API key to access to their functionality. However, it is worth noting that an API key does not identify a user nor a unique instance of an application. Moreover, API keys are simply not authoritative, as the key itself is typically obfuscated in compiled code and so subject to extraction by any skilled developer. Thus, using API keys as credentials can result in identity-based attacks; such as identity theft.

For instance, if the interface of the Domain orchestration service or the E2E service orchestration service is not secured, an attacker may gain access to capabilities for the network and service management without authorization. The attacker may change configuration of E2E service instance to fail its SLA. The attacker may also create E2E service instances demanding significant network resources in order to exhaust the network resources and potentially occasion the network unavailability.

3) Man-In-The-Middle attacks: Unencrypted transmission of API messages introduces the risk of Man-In-The-Middle (MITM) attack where an attacker is secretly positioned between the API consumer and provider. The attack leads to the interception of API messages, which can reveal confidential information, as well as to the their manipulation.

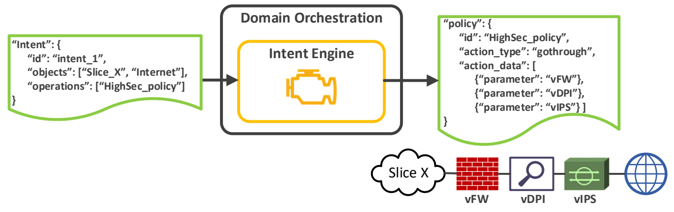
4) (Distributed) Denial-of-Service attacks: APIs are potentially vulnerable to Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks that can make an API out of order by submerging it with a massive amount of requests.

B. Intent-based Interfaces' Security Threats

The use of Intent-based technologies is identified as a mean for automation [3]. Moreover, Intent-based interfaces are introduced as one of the ZSM architecture's principles [1]. Indeed, Intent-based interfaces expose high-level abstractions allowing to decouple from underlying technology- and vendor-specific details and hide complexity from the user. They specify policies rather than mechanisms of how to realize them. Thus, Intents are about "What", not "How". Decoupling the How and What of the service gives the controller/orchestrator more freedom in design and implementation choices. Note that JSON is one of the candidates for autonomic network Intent format. The potential threats related to use of Intent-based interfaces include *information exposure*, *undesirable configuration* and *abnormal behavior*.

1) Information exposure: Intents can vehicle information about the desires of the application such as connecting with peers, advertising services or content, and regulating network traffic. Thus, intercepting such information by an unauthorized entity can compromise system security objectives (e.g., privacy, confidentiality) and result in launch of other attacks.

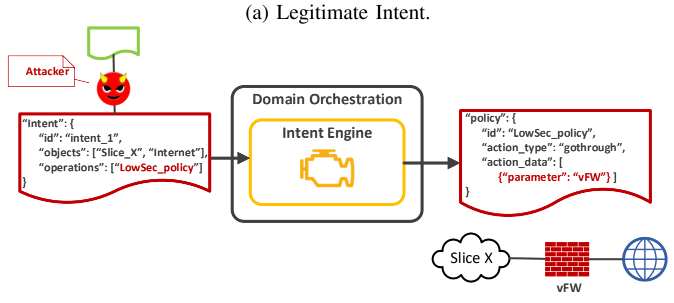
2) Undesirable configuration: The workflow sent to a "Domain orchestration service" is defined in a domain service model (i.e., blueprint) which can be consumed. The domain service model describes all required infrastructure resources and consumed services, their configuration, their topology, their policies and their placement in the network [1]. Some aspects of the service models can be intent-based and require a mapping of the intent (e.g., a security level) to one or more orchestration actions (e.g., security VNFs selection and chaining). For example, a high level of security is mapped to a service chaining of a firewall, a DPI, and IPS VNFs as illustrated in Fig. 2a. Meanwhile, a low level of security is translated to instantiation of only a firewall VNF. Let us assume a scenario where the following intent-based service model is sent to a "Domain orchestration service": "HTTP traffic from slice X to Internet has a high security level". Thus, if an attacker maliciously tampers the Intent by changing the security level from "high" to "low" (see Fig. 2b), an undesirable security level will be set to slice X, making the slice vulnerable to security threats.



```
"Intent": {
    "id": "intent_1",
    "objects": ["Slice_X", "Internet"],
    "operations": ["HighSec_policy"]
}
```

```
"policy": {
    "id": "HighSec_policy",
    "action_type": "gothrough",
    "action_data": [
        {"parameter": "vFW"},
        {"parameter": "vDPI"},
        {"parameter": "vIPS"} ]
}
```

(a) Legitimate Intent.



```
"Intent": {
    "id": "intent_1",
    "objects": ["Slice_X", "Internet"],
    "operations": ["LowSec_policy"]
}
```

```
"policy": {
    "id": "LowSec_policy",
    "action_type": "gothrough",
    "action_data": [
        {"parameter": "vFW"} ]
}
```

(b) Tampered Intent.

Fig. 2: Intent Tampering Illustrative Example.

3) Abnormal behavior: A malformed Intent sent to a "Domain orchestration service" may result in an abnormal behavior from this service during the Intent rendering. For instance, the "Domain orchestration service" may be aborted or rebooted, leading to Denial of Service (DoS). Alternatively, the "Domain orchestration service" may map the intent to a policy that could result in network mis-configuration, causing network outage or security vulnerabilities.

C. Security threats driven by Closed-Loop Networked Automation

The Closed-Loop management automation is a feedback-driven process that seeks to achieve a set of objectives (e.g., self-optimization, enhanced use of network resources, and automated service fulfillment and assurance) without further human intervention. The pairing of feedback loops with telemetry data, analytics and orchestration services empowers intent-based network services.

The closed loop management automation is one of the ZSM architecture's principles. Fig. 3 depicts a closed-loop example based on ZSM system. The example shows a legitimate scenario where a fault event collected by the "Domain Data Collection" triggers the closed-loop process by publishing the fault event, which will be consumed by the "Domain Intelligence". This last determines that the link is bad and requests the "Domain Control" to change the router configuration to reroute the traffic.

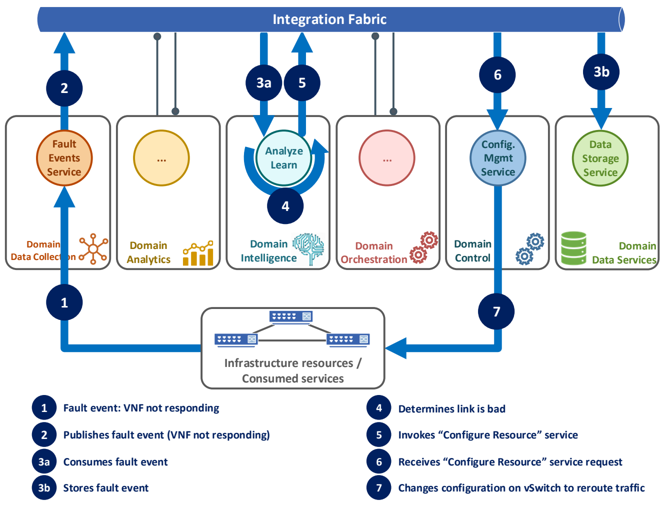


Fig. 3: Closed-Loop Illustrative Example.

The networked closed-loop control system is prone to several security threats, such as DoS, MITM attacks, and Deception attacks. An attacker can influence the communication channels in the control system in order to observe, hide, create or change information in the attacked network channel. A deception attack intends to tamper transmitted data packages, by replaying e.g. past data, causing false feedback information. For example, an attacker may send a fake fault event to the "Domain Data Collection" telling that "VNF X is not responding" (even if it is not). As part of the closed loop at the domain level, the "Fault events service" of the "Domain Data Collection" publishes the faked fault event, which will be consumed by the "Domain Intelligence" services. The VNF unresponsiveness may be due to several causes:

- The "Domain Intelligence" may determine that the link is bad and then a "Configure Resource" query is made to "Domain Control" to change the switch configuration for traffic rerouting. If the attacker can hijack the response of that query to reroute the traffic via an attacker-controlled switch, the attacker's switch can then act as a Man-In-The-Middle. This scenario is illustrated in Fig. 4.

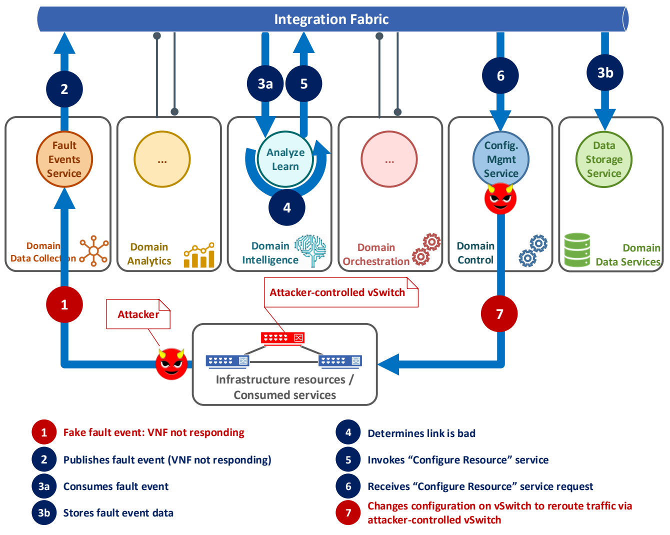

Integration Fabric
Fault Events Service
...
Analyze Learn
...
Config. Mgmt Service
Data Storage Service
Domain Data Collection
Domain Analytics
Domain Intelligence
Domain Orchestration
Domain Control
Domain Data Services
Attacker
Attacker-controlled vSwitch
Infrastructure resources / Consumed services
1 Fake fault event: VNF not responding
2 Publishes fault event (VNF not responding)
3a Consumes fault event
3b Stores fault event data
4 Determines link is bad
5 Invokes "Configure Resource" service
6 Receives "Configure Resource" service request
7 Changes configuration on vSwitch to reroute traffic via attacker-controlled vSwitch

Fig. 4: Attack driven by Closed-Loop Automation.

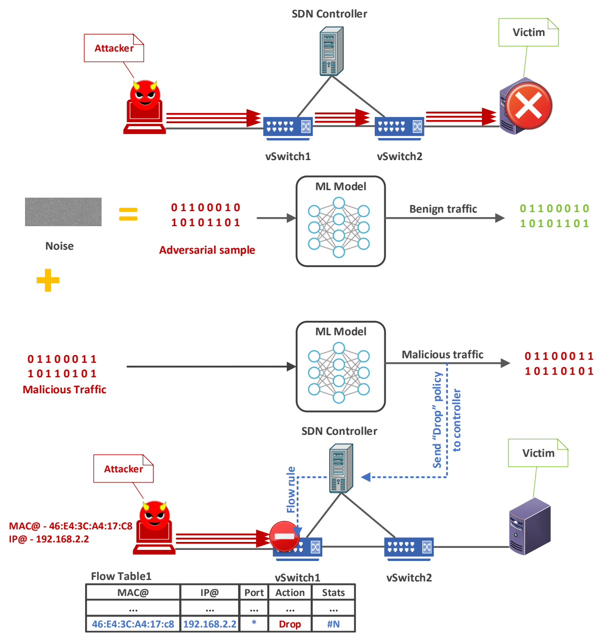
- The "Domain Intelligence" may determine that the VNF is under heavy load and then a query is sent to "Domain orchestration" services to scale the VNF's resources or to instantiate new VNFs to handle the increased load. In both cases, more resources will be allocated to this network function, which can lead to exhaustion of infrastructure resources (i.e., DoS).

D. *AI/ML-based Attacks*

The combination of an intent-driven approach and AI/ML techniques for network and service management will result in substantial increases in service effectiveness and efficiency through objectives fulfillment in optimal and reduced management time and in functionality by unlocking new business models. AI/ML techniques play an important role in empowering functionalities such as self-planning, self-optimization, self-healing, and self-protecting. For instance, the use of Deep Learning (DL) is recently gaining momentum in enabling intelligent network management and operation capabilities, such as traffic classification, traffic forecasting, mobility prediction, resource allocation, and network security [4]. However, the growing enthusiasm for AI/ML adoption in the management of next-generation networks could be dampened if security concerns related to use of AI/ML techniques are not tackled. In fact, the use of AI/ML and other data analytics technologies introduces new attack vectors. Barreno et al. [5] provided a taxonomy of attacks targeting the training phase (i.e., poisoning attacks), as well as the test phase (i.e., evasion attacks). In poisoning attacks, an attacker focuses on tampering the training data, by injecting carefully crafted malicious samples, to influence the learning outcome. Meanwhile, an evasion attack attempts to bypass the learned model by introducing small perturbations to the test instances. Such perturbations are called adversarial examples. Recently, new attacks have emerged, particularly targeting ML-as-a-Service (MLaaS), namely: model inversion attacks [6] and model extraction attacks [7]. Leveraging the outputs of the targeted ML model, the former aims to infer the training data, while the latter focuses on stealing the model parameters to reproduce a (near)-equivalent ML model. The attacks may violate training data privacy or facilitate evasion attacks. Biggio *et al.* [8] and Fredrikson *et al.* [7] showed that neural networks are prone to evasion attacks and model inversion attacks, respectively. Deep neural networks can be exposed to model extraction attacks [7]. Several studies (e.g., [6], [7]) proposed successful model extraction and inversion attacks against MLaaS platforms, such as Amazon ML, Microsoft ML and BigML. Pattanaik *et al.* [9] demonstrated the vulnerability of Deep Reinforcement Learning to adversarial attacks. It has also been shown in [10] that Federated Learning is vulnerable to poisoning attacks.

Attacks against ML can be classified according to three properties, namely [5]: influence, specificity and security violation. The *influence* property indicates the attacker's capabilities. In this case, an attacker can launch causative attacks if he/she is able to influence the training data, or exploratory attacks if he/she is only able to tamper the learning outcomes. The *specificity* relates to the attack's target. Thus, an attack is targeted if it focuses on the misclassification of a specific sample or class of samples, while it is indiscriminate if it aims to break down the trained model's performance by misclassifying any sample. From *security violation* perspective, an attacker may aim to cause either integrity, availability or privacy violation. Integrity attacks result in indecision, delayed decisions, poor or even wrong decisions [11]. However, availability attacks seek to considerably increase the classification errors that the system becomes effectively unusable. The privacy attacks aim to obtain private information about the system, its users or data by reverse-engineering the learning algorithm. The attacks against ML can be also divided into two categories based on the attacker's knowledge [12], namely: (i) White-box attacks, which assume that the attacker has complete knowledge about the training data, the algorithm and its hyper-parameters. (ii) Black-box attacks, which assume that the attacker has no knowledge about the algorithm and its hyper-parameters. In this case, the attacker first observes the ML-based system response to its query and then uses this outcome to craft adversarial examples. Note that the ML transferability property allows to convert white-box adversarial attacks into black-box attacks.

The ZSM's E2E service intelligence provide services that enable both decision-making and predictions capabilities [1]. The decisions making leverage information obtained from domain data collection services and common data services. Thus, an adversary may craft inputs to fool the ML model used by the E2E service intelligence services into making wrong decisions or predictions, potentially resulting in performance degradation and financial loss, as well as endangering SLA fulfillment and security guarantees. For instance, an adversary can inject crafted samples that let the E2E service intelligence services wrongly forecast the future resource requirements of an E2E service, or to trigger an inappropriate management policy (e.g., reconfiguration, scaling) of an E2E service. In another attack scenario, an adversary may generate crafted samples that result in misclassifying a malicious or anomalous traffic as normal traffic. The illustrative example depicted in Fig. 5 shows how a ML model trained to detect malicious traffic generated by a DoS attack can be fooled by adversarial samples to wrongly identify the malicious traffic as benign traffic, allowing the DoS attack to succeed. Note that domain intelligence services are prone to the same attacks.



MAC@	IP@	Port	Action	Stats
...	...	...	...	...
46:E4:3C:A4:17:c8	192.168.2.2	*	Drop	#N

Fig. 5: Illustrative Example of an Adversarial Attack against ML Model.

E. Attacks due to Adoption of Programmable Network Technologies

The ZSM framework is based on the programmable networking approach by leveraging SDN and NFV technologies. While SDN and NFV technologies offer benefits to create automated, scalable and customizable networks, they unleash new attack vectors. Open Network Foundation (ONF) [13] identified the security threats associated to the different SDN layers (i.e., control, data and application layers). The identified threats were classified into six categories, namely spoofing (e.g., openflow switch and/or controller impersonation), tampering (e.g., configuration data), repudiation, information disclosure, DoS, and privilege escalation, following Microsoft STRIDE model. Yoon *et al.* [14] have listed 22 attack vectors (e.g., DoS, malformed control message injection, unauthorized network-view manipulation, unauthorized network control, and eavesdropping) stemming from the design and implementation weaknesses of the existing SDN controller platforms. In the other side, ETSI [15] has identified the threat surface of NFV as the union of generic virtualization threats (e.g., memory leakage, interrupt isolation), generic networking threats (e.g., flooding attacks, routing security), and the threats due to combining virtualization technology with networking (e.g., introspection attacks). As software components, VNFs are vulnerable to software vulnerabilities including design, implementation and configuration flaws. Thus, compromised VNFs could provide wrong monitoring data that can mislead the analytics and intelligence services in ZSM framework.

IV. Potential Mitigation Mechanisms

In this section, we present the potential mitigation measures and best practices that should be adopted to make a ZSM system resilient to the aforementioned security threats. We also highlight some research directions.

A. API Security

API security is a cornerstone to ensure that only envisaged consumers (e.g., digital storefront or services in the MD, the Common Data Services and the E2E Service MD) are allowed to interact with ZSM's APIs in an authorized way. To this end, several security measures can be leveraged, including authentication, authorization, communication encryption and input validation. Authorization can be empowered using OAuth2.0 or JWT tokens, which allow to achieve least privilege requirement. Role Based Access Control (RBAC), Attribute Based Access Control (ABAC) and Access Control Lists (ACLs) are also recommended to enforce fine-grained authorization that limit access to APIs and operations in the API. Signed JWT tokens and OpenID Connect are means to accomplish authentication. API messages must be encrypted to prevent manipulation, eavesdropping and Man-In-The-Middle (MITM) attacks. Transport Layer Security (TLS) protocol, with at least version 1.2 and using certificates compliant with IETF RFC 5280, is recommended. TLS provides integrity and confidentiality of APIs messages including access tokens. When associated with certificates, it also provides consumer-side authentication. All data within the URL, query parameters, HTTP headers and content of ZSM APIs should be properly validated to thwart injection attacks. As Open APIs are based on JSON format, the input validation of JSON schema should also be enforced to prevent unknown parameters from being exposed to ZSM services. The mitigation of brute force and API-specific (D)DoS attacks can be achieved through throttling/rate limiting the usage of APIs. As a result, API requests will be blocked when the usage rate limits are reached, allowing the availability of services. A common practice to provide the API security measures recommended above is the use of API gateways and microgateways to enforce north-south and east-west security, respectively [2]. The ZSM's Intra- and Inter-domain integration fabrics can be used to deploy the API gateways and microgateways.

The growing number of services, the variety of APIs and the high volume of API traffic in an envisioned ZSM system will make the identification and mitigation of API threats using traditional measures a complex and inefficient task. To overcome this limitation, a new direction to complement and strengthen the aforementioned security capabilities is the application of AI and ML techniques. Indeed, AI-based API security provides proactive monitoring and detection of API threats and empower their automatic mitigation. However, the integration of an AI engine in an API security solution will bring its own vector attacks as we already mentioned in the previous section.

B. Intent Security

The information passed by an intent-based interface is sensitive and consequently have to be accessed and manipulated only by authorized entities. Authentication (e.g., OpenID Connect, signed JWT tokens) and authorization (e.g., OAuth2.0, RBAC) mechanisms are needed to ensure, respectively, mutual authentication between intent producer and intent consumer, and controlled access to intent-based interface. The exchange of intents through a secure transport protocol, such as TLS 1.2, is necessary to foster both intent integrity and confidentiality, and consequently prevent Intent tampering and sniffing, respectively. To deal with malformed and/or conflicting intents, Intent engine should provide the capabilities to validate the Intent format and detect/resolve potential conflict situations.

C. AI/ML Security

Adversarial Machine Learning (AML) [16] is an emerging research discipline that aims to make ML techniques resilient to adversarial attacks. It is intended for assessing the vulnerability of ML algorithms to attacks and designing appropriate countermeasures which yield more robust learning. Potential defenses against adversarial attacks include input validation, adversarial training, defensive distillation, defense Generative Adversarial Networks (GANs) and concept drift. Input validation consists in validating the input samples before they are fed into the ML model. In adversarial training, the model is explicitly trained on adversarial examples in order to learn how to resist them. Defensive distillation is the process of using knowledge derived from an ML model to enhance its own robustness to adversarial samples. While both adversarial training and defensive distillation have proven their effectiveness against white-box attacks, they fail to counter black-box attacks. Defense GANs seek to denoise adversarial examples by projecting input samples onto the range of the GAN's generator before injecting them to the ML model. It is worth noting that defense GANs are resilient to both white-box and black-box attacks. Concept drift faces adversarial attacks by detecting drop in ML model performance. To mitigate model inversion and model extraction attacks, various solutions have been proposed, ranging from restricting information provided by ML APIs, adding noise to the ML predictions, to adding noise to execution time of the ML model. While AML has attracted much interest in computer vision field, only very few contributions (e.g., [12], [11]) have addressed ML security in the context of service and network management. Usama *et al.* [12] highlight the importance of tackling adversarial attacks against cognitive self-organizing networks. As a proof of concept, white-box evasion attacks against Convolutional Neural Network were designed to demonstrate how a malware classifier can be escaped. Han *et al.* [11] explored the resistance of Reinforcement Learning (RL) to different forms of causative attacks in the context of autonomous cyber-defense in SDNs. Realizing the importance of AI/ML security, ETSI has very recently initiated a new Industry Specification Group on Securing Artificial Intelligence (ISG SAI) [1]. The group aims to develop technical specifications to mitigate threats caused by the deployment of AI in ICT field. The security of ML models is paramount to their integration in a service and network management platform for next-generation networks. Thus, more research efforts need to be devoted to master how adversarial attacks can be launched and countered in networking environment. Another research direction is to propose a certification framework to assess the security properties of ML techniques.

D. SDN/NFV Security

Several solutions (e.g., [13], [14]) have been recommended to mitigate attacks against SDNs. Mutual authentication should be enforced to prevent impersonation of SDN applications, SDN controllers and switches. Authorization mechanisms (e.g., RBAC) are necessary to control the access level and avoid privilege abuse which is a primary cause for various attacks, such as DoS and data leakage. Confidentiality and integrity of messages exchanged on communication channels (i.e., North-South and East-West interfaces) must be preserved to prevent information disclosure and tampering. To this end, encryption, digital signature, and Message Authentication Code (MAC) algorithms could be leveraged. Tamper-proof devices, such as Trusted Platform Module (TPM), are recommended to shelter sensitive data, encryption keys, password and certificates. To tackle DoS/DDoS attacks and guarantee availability, potential mitigation solutions include, among others, malicious traffic monitoring, limiting the number of flow requests, resource usage monitoring and restrictions, using distributed SDN controller architecture.

To deal with NFV security issues, ETSI NFV Security Group comes out with a set of best practices and recommendations [17]. TPM and virtual TPM are potential enablers to foster VNF boot integrity and empower remote attestation. To establish VNF confidentiality and integrity at run time, Trusted Execution Environments (e.g., Intels Software Guard Extensions (SGX), AMDs Secure Encrypted Virtualization (SEV)) are acknowledged as a promising solution. Indeed, TEEs are hardware-based solutions that have been specifically designed for providing total integrity and confidentiality even in the presence of a high privileged malicious operator or a malicious hypervisor. Traffic monitoring and filtering using Intrusion Detection System (IDS) and firewalls is a key technique to detect and mitigate DoS/DDoS attacks. Resource isolation and usage limitation are further measures to guard against DoS/DDoS attacks based on resource exhaustion. Strong authentication and authorization mechanisms are required to thwart NFV MANO hijacking attack. The establishment of secure communication channel between NFV MANO and NFVI is mandatory to defeat traffic eavesdropping and modification, as well as MITM attacks. The exploitation of software vulnerabilities, brought on by NFV components (i.e., VNFs, VMs, containers,

[1]https://www.etsi.org/committee/1640-sai

hypervisors, etc.), can be countered by establishing secure software patching procedures and applying recommended system hardening techniques, such as removing unnecessary services, customizing default configurations, enabling OS-level access and confinement controls (e.g., SELinux, sVirt).

The main security threats discussed above as well as their potential mitigation measures are summarized in Table I.

Enabler	Security Threats	Mitigation Measures
Open API	Parameter attacks	Input validation
	Identity attacks Tampering attacks MITM attack	Authentication and authorization controls Secure communication
	(D)DoS	Throttling/rate limiting the usage of APIs
Intent-based Interfaces	Information exposure Intent tampering	Authentication and authorization controls Secure communication
	Malformed Intent	Intent format validation
	Conflicting Intents	Conflict detection/resolution
AI/ML	Adversarial attacks	Input validation Adversarial training Defensive distillation Defense GANs Concept drift
	Model extraction attacks Model Inversion attacks	Control information provided by ML APIs Add noise to ML prediction Add noise to execution time of the ML model
SDN/NFV	Spoofing Privilege escalation Information disclosure Tampering	Authentication and authorization controls Secure communication TPM, vTPM
	DoS	Malicious traffic monitoring Limiting the number of flow requests Resource monitoring and usage limitation Resource isolation Distributed SDN controller architecture
	Introspection Attacks	TEE
	Software vulnerabilities	Secure software patching procedures System hardening techniques

TABLE I: ZSM security threats and mitigation measures.

After discussing the potential countermeasures that should be provided to mitigate the possible security threats brought by the ZSM's enablers, we propose a security-enhanced ZSM architecture where the key security functional components are introduced into the ZSM reference architecture to empower the security of a ZSM system as a whole. Fig. 6 shows the recommended security-enhanced ZSM reference architecture at a management domain level, including the E2E service management domain. Integrating the API gateway/microgateway ensures that management services and data are consumed only by authenticated and authorized parties. Moreover, it prevents parameter attacks by validating the API's content. All communication between ZSM components and/or with domain managed infrastructure resources are encrypted, enabling protection of data in motion. To handle model extraction and inversion attacks against ML models, we suggest to incorporate an "ML APIs Control' component at the API gateway/microgateway offering mitigation capabilities, such as restricting information provided by ML APIs. Data encryption and integrity capabilities should be provided by data services to guarantee the protection of data at rest. The "policy management" supporting service recommended in the ZSM reference architecture can play the role of the "intent engine" to handle the intent malformation and conflicts problems. Following the vision of a ZSM system, the attacks against the ZSM's components (i.e., management functions) and data should be proactively and autonomously detected and mitigated. To this end, security-related analytics and intelligence services should be provided. For instance, the anomalies in AI/ML model performance can be monitored by the "AI model performance evaluation" service. This service can be leveraged by the "AI model assessment" service to take the most appropriate decision in order to mitigate the detected anomaly. Similarly, it is crucial to introduce "API analytics" and "AI engine for API security" services into domain analytics and domain intelligence, respectively, in order to detect security anomalies against APIs. Those services can be leveraged for instance by the "throttling/rate limiting" component to mitigate the (D)DoS attacks against APIs. To close the loop, the security functions related to ZSM's components and data are managed by the "domain security orchestration" services in compliance with the defined security policies.

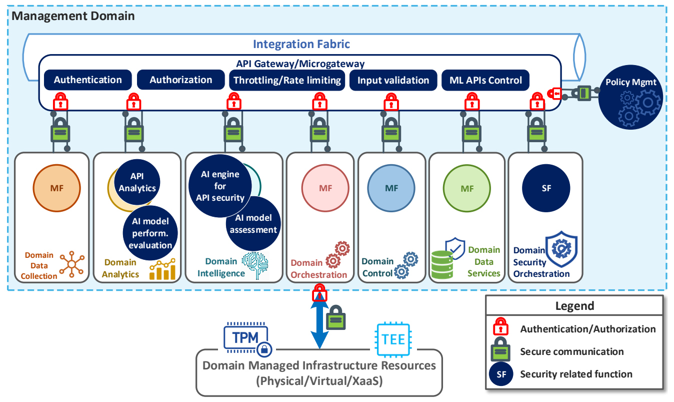


Fig. 6: Security-Enhanced ZSM Reference Architecture.

V. Domain Intelligence Security: Practical Study of Attacks and Defenses.

Considering the key role that AI/ML techniques will play in empowering the intelligent full ANSMO envisioned by ZSM, this section presents a practical study demonstrating the vulnerability of Deep Learning (DL) models to adversarial attacks as well as the effectiveness of adversarial learning defense. To this end, a DL-based DoS detection model is built using Multi Layer Perceptron (MLP) algorithm. The proposed model is trained on the recent intrusion detection dataset, CICIDS2017 [2], where only network flows corresponding to normal traffic and DoS/DDoS attacks are used. 70% of the dataset's flows are used to train the model and the remaining 30% flows are used as a test set to assess the model's performance on unseen data. The model is implemented using the Pythons DL library Keras running on a TensorFlow backend. It achieved an accuracy of 99.65% on the test set.

The attacker is assumed to have a full-knowledge (i.e., white-box attack) on the targeted model to generate adversarial flows that will be misclassified by the model (i.e., (D)DoS flow classified as normal flow or vice versa). Three attacks are considered: Fast Gradient Sign Method (FGSM), Basic Iterative Method (BIM) and Jacobian-based Saliency Map Attack (JSMA). FGSM generates adversarial examples by

[2]https://www.unb.ca/cic/datasets/ids-2017.html

performing a one step gradient update in the direction of the gradient's sign of the loss function relative to the input. The input is than altered by adding a perturbation that can be expressed as: $\eta = \epsilon.sign(\nabla_x J(\theta, x, y))$, where x is a sample (i.e., network flow), y is the label of x, $J(\theta, x, y)$ is the loss function used to generate the adversarial example and ϵ is the perturbation magnitude. BIM extends FGSM by applying it multiple times with small step size and clipping the values of the adversarial example after each step such that they are within an ϵ-neighborhood of the original sample. JSMA creates an adversarial example by perturbing the minimal number of features of the original sample based on Saliency map. The attacks are carried out using the Cleverhans library [3] to craft adversarial examples from the test set. We investigated the effectiveness of adversarial training to counteract the implemented attacks. The adversarial training is performed based on adversarial examples generated using FGSM and BIM attacks. Fig. 7 shows the accuracy of the original model and its adversarially trained variant on adversarial flows for different values of ϵ. For the three attacks, we observe a significant drop in the original model's accuracy that is above 58% and can reach 86.83% under BIM attack. It is worth mentioning that the advantage of JSMA over FGSM and BIM lies in reducing the number of perturbed features, making the generation of adversarial network flows more feasible. In fact, only 14% of features on average are modified by JSMA compared to 78% for FGSM and BIM. The results show also that the model's robustness is significantly improved when the model is trained with adversarial examples generated by the same attack (i.e., FGSM-FGSM, BIM-BIM). In the case of JSMA, we notice that BIM-based adversarial training (i.e., JSMA-BIM) yields a more robust model to the JSMA attack.

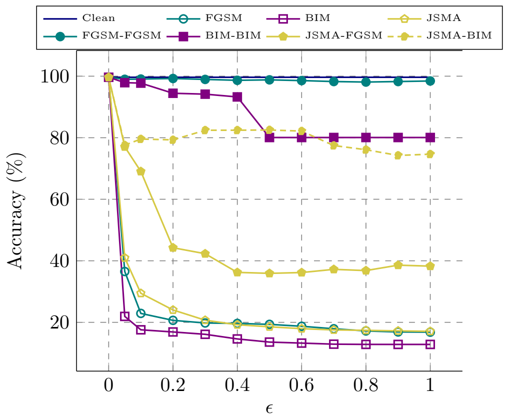


Fig. 7: The accuracy of DL-based DoS detection model and its adversarially trained variant under FGSM, BIM and JSMA attacks.

VI. Conclusion

This paper identified the threat surface that may be introduced by a ZSM system. To enable full ANSMO, the ZSM framework relies on several emerging enablers (e.g., SDN/NFV, AI/ML and model-driven interfaces), bringing new security threats and broadening the attack surface. We shed lights on potential security risks that may hinder the realization of the full ANSMO vision. We then recommended several countermeasures to resist these attacks, considering existing security solutions or advising the development of new ones. Needless to say, AI will be a game-changer for advancing management operations and providing efficient resiliency to attacks in next-generation networks. Nevertheless, the adoption of AI is challenged by properly addressing its own security issues. Thus, investigating the AI's capability in building up effective mitigation measures and tackling the AI security issues in a networking environment are potential future research topics.

[3] https://github.com/tensorflow/cleverhans

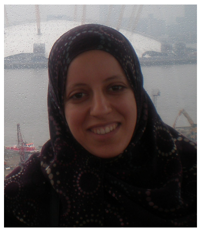
Acknowledgment

This work was supported in part by the Academy of Finland Project 6Genesis Flagship (Grant No. 318927), CSN (Grant No. 311654) and the European Unions Horizon 2020 research and innovation programme under the INSPIRE-5Gplus project (Grant No. 871808).

References

[1] ETSI GS ZSM 002, "Zero-touch Network and Service Management (ZSM); Reference Architecture," Aug. 2019.
[2] Gartner, "How to Build an Effective API Security Strategy," Dec. 2017.
[3] DRAFT ETSI GR ZSM-005, "Zero Touch Network and Service Management (ZSM); Means of Automation," Dec. 2018.
[4] C. Zhang, P. Patras, and H. Haddadi, "Deep Learning in Mobile and Wireless Networking: A Survey," *IEEE Communications Surveys & Tutorials*, March 2019.
[5] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. Tygar, "Can Machine Learning Be Secure?" in *In Proc. of ASIACCS06*, 2006, pp. 16 – 25.
[6] M. Fredrikson, S. Jha, and T. Ristenpart, "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures," in *In Proc. of the ACM SIGSAC Conference on Computer and Communications Security (CCS'15)*, Oct. 2015, pp. 1322 – 1333.
[7] F. Tramr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing Machine Learning Models via Prediction APIs," in *In Proc. of the 25th USENIX Security Symposium*, Aug. 2016, pp. 601 – 618.
[8] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Srndic, P. Laskov, G. Giacinto, and F. Roli, "Evasion Attacks Against Machine Learning at Test Time," in *In Machine Learning and Knowledge Discovery in Databases, Part III, volume 8190 of LNCS*, 2013, pp. 387 – 402.
[9] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary, "Robust Deep Reinforcement Learning with Adversarial Attacks," in *In Proc. of the 17th International Conf. on Autonomous Agents and MultiAgent Systems (AAMAS'18)*, July 2018, pp. 2040 – 2042.
[10] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Model Poisoning Attacks in Federated Learning," in *In Workshop on Security in Machine Learning (SecML), collocated with the 32nd Conference on Neural Information Processing Systems (NeurIPS'18)*, Dec. 2018.
[11] Y. Han, B. I. Rubinstein, T. Abraham, T. Alpcan, O. De Vel, S. Erfani, D. Hubczenko, C. Leckie, and P. Montague, "Reinforcement Learning for Autonomous Defence in Software-Defined Networking," in *In Proc. of the 9th International Conf. on Decision and Game Theory for Security (GameSec)*, Aug. 2018, pp. 145 – 165.
[12] M. Usama, J. Qadir, and A. Al-Fuqaha, "Adversarial attacks on cognitive self-organizing networks: The challenge and the way forward," *CoRR*, vol. abs/1810.07242, 2018.
[13] Open Networking Foundation, "Threat Analysis for the SDN Architecture," *TR-530, Version 1.0*, July 2016.
[14] C. Yoon, S. Lee, H. Kang, T. Park, S. Shin, V. Yegnesw, P. Porras, and G. Gu, "Flow Wars: Systemizing the Attack Surface and Defenses in Software-Defined Networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 6, pp. 3514 – 3530, Dec. 2017.
[15] ETSI GS NFV-SEC 001, "Network Functions Virtualisation (NFV); NFV Security; Problem Statement," Oct. 2014.
[16] L. Haung, A. D. Joseph, B. Nelson, B. I. Rubinstrein, and J. D. Tygar, "Adversarial Machine Leaning," in *In Proc. of 4th ACM Workshop on Artificial Intelligence and Security,*, Oct. 2011, pp. 43 – 58.

[17] ETSI GS NFV-SEC 003, “Network Functions Virtualisation (NFV); NFV Security; Security and Trust Guidance,” Dec. 2014.



Chafika Benzaïd is currently a PostDoc researcher at MOSA!C Lab, Aalto University. She is an associate professor and research fellow in Computer Science Department at University of Sciences and Technology Houari Boumediene (USTHB). She obtained her PhD degree in Computer Sciences from USTHB in 2009. Her current research interests include AI-driven network security and AI security. She serves/served as a TPC member for several international conferences and as a reviewer for multiple international journals.

Tarik Taleb is Professor at Aalto University. He is the founder and director of the MOSA!C Lab (www.mosaic-lab.org). Prior to that, he was a senior researcher and 3GPP standards expert at NEC Europe Ltd., Germany. He also worked as an assistant professor at Tohoku University, Japan. He received his B.E. degree in information engineering, and his M.Sc. and Ph.D. degrees in information sciences from Tohoku University in 2001, 2003, and 2005, respectively.