INVITED PAPER *Special Section on Network Virtualization and Network Softwarization for Diverse 5G Services*

Towards Autonomous Security Assurance in 5G Infrastructures

Stefan COVACI[†a)], **Matteo REPETTO**[††], ***and*** **Fulvio RISSO**[†††], ***Nonmembers***

SUMMARY 5G infrastructures will heavily rely on novel paradigms such as Network Function Virtualization and Service Function Chaining to build complex business chains involving multiple parties. Although virtualization of security middleboxes looks a common practice today, we argue that this approach is inefficient and does not fit the peculiar characteristics of virtualized environments. In this paper, we outline a new paradigm towards autonomous security assurance in 5G infrastructures, leveraging service orchestration for semi-autonomous management and reaction, yet decoupling security management from service graph design. Our work is expected to improve the design and deployment of complex business chains, as well as the application of artificial intelligence and machine learning techniques over large and intertwined security datasets. We describe the overall concept and architecture, and discuss in details the three architectural layers. We also report preliminary work on implementation of the system, by introducing relevant technologies.

key words: *cyber-security, NFV, 5G, service chaining*

1. Introduction

Beyond continuous improvement in key communication performance indexes (like bandwidth, delay, jitter), the most disruptive evolution of fifth-generation mobile networks (5G) will certainly be the massive deployment of computing and storage resources in all network segments. This will transform legacy networks into pervasive and capillary orchestration platforms [1], [2], hence enabling new services and increasing the agility of the infrastructure [3]. The extensions to the very edge of the network (e.g., *fog* and *edge computing*) also promise new management and connectivity models to effectively integrate the *Internet of Things* (IoT) and requires new orchestration paradigms [4]. This enables to tackle even the most challenging requirements from industries that, for various reasons, have not yet adopted cloud technologies (e.g., eHealth, factory automation, automotive and mobility, energy) [5].

Despite the considerable evolution and progress in *virtualization* and *orchestration*, which now effectively support (semi-)autonomous deployment and life-cycle management of even complex business chains over large distributed computing environments [6], security has not evolved at the same pace. Multi-tenancy and the coexistence of many diverse end-to-end 5G services make the common security perimeter model obsolete, and demand for new paradigms that could effectively identify vulnerabilities and detect attacks in novel computing paradigms [7]. Indeed, de-coupling software from the underlying hardware brings immediate benefits in terms of elasticity, portability, automation, and resiliency, but the intermediate virtualization tier also raises new security concerns about the mutual trustworthiness between the two layers. Beyond logical isolation between tenants, there is anyway a tight security relationship among the infrastructure and the diverse services: untrusted or compromised hardware may eventually vanish even the most secured applications; in a specular manner, compromised software may be exploited to leverage vulnerabilities in the hypervisor to gain access to the physical infrastructure [8].

A common trend today is the implementation of software versions of legacy security appliances, largely motivated by the prominence of the Infrastructure-as-a-Service cloud model and its undeniable similarity with physical environments. We argue that this approach is inefficient, increases the attack surface, does not effectively tackle complex multi-vector attacks, and creates overlapping and conflicts between service developers and security staff.

Motivated by the lack of common and uniform Security-as-a-Service models in existing Network Function Virtualization (NFV) [9] and the substantial untrustworthiness between software functions and the underlying infrastructure, we advocate in this paper tighter integration of security into software orchestration. In other words, security policies should not be left in the hands of final users or service developers, who are more interested in the semantic of the service and may lack the interest to properly protect the service itself. Instead, security is provided automatically as a part of the orchestration software, which takes care of hardening the service, and properly report its real-time status (with respect to security aspects) to the different stakeholders. Therefore, in this paper we address the need for systematic and programmatic security awareness, by de-coupling inspection tasks from the detection logic; the former is to be integrated into the different forms of virtualization containers, and the latter to be part of the orchestration process and directly interacting with the application management to provide situational awareness and to support quick reaction and mitigation actions (see Fig. 1).

The rest of the paper is organized as follows. Section 2 reviews the main motivations behind the need for a novel paradigm for cybersecurity assurance in 5G. Section 3 describes an example of business chain which requires in-

Manuscript received July 23, 2018.
Manuscript revised August 6, 2018.
Manuscript publicized September 20, 2018.
[†]The author is with Technical University Berlin, Germany.
[††]The author is with CNIT, Italy.
[†††]The author is with Politecnico di Torino, Italy.
a) E-mail: stefan.covaci@tu-berlin.de
DOI: 10.1587/transcom.2018NVI0001

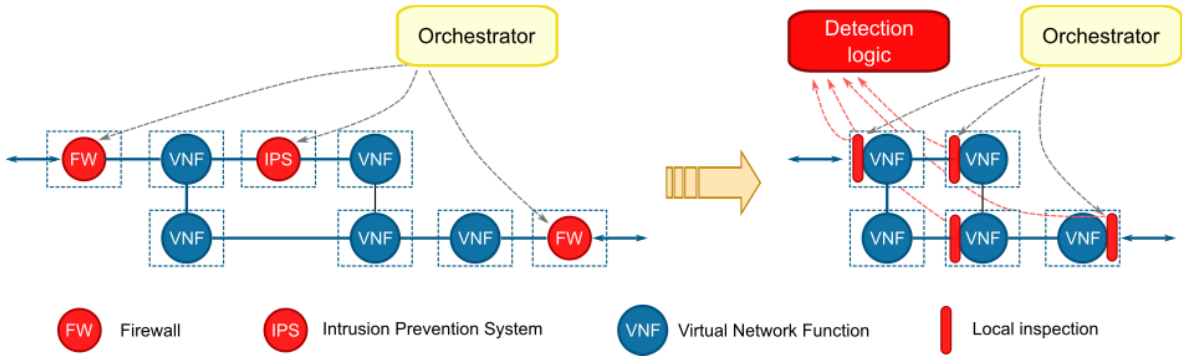

Fig. 1 A transition from software implementation of security appliances to a novel framework with local monitoring and centralized detection logic.

tegration between services deployed and owned by different parties. We introduce a novel security concept in Sect. 4, then we discuss in details the components for detection and mitigation of cyber-attacks in Sect. 5. We give preliminary indication about the implementation of the framework in Sect. 6. Finally, we give our conclusion in Sect. 7.

2. Motivations for a New Paradigm

End-to-end services in 5G will be implemented as flexible and elastic chains of Virtual Network Functions (VNFs) and business functions, deployed over a heterogeneous mix of clouds, software-defined wide area networks (SD-WAN), edge installations, and even smart things. The creation of secure service chains will require proper solutions for identity management, attestation, authentication, and encryption of both software and hardware in 5G deployments.

In the following we will provide the main motivations for a new paradigm in handling security.

Run-time monitoring and security assurance. The ETSI NFV ISG [10] has already addressed many security implications of deployments on virtualised infrastructures but has not yet taken into account the integrity of a running system beyond its initial stages of provisioning, boot and loading software [11]. Behavioural monitoring of host systems, virtual services, and network traffic is anyway unavoidable to build wide situational awareness over single and multiple services, as well as the overall critical infrastructure.

Novel orchestrators with automatic provisioning and monitoring of security services. Software implementations of security appliances (Intrusion Prevention/Detection Systems, Firewalls, Antivirus, Network Access Control, etc.) may be easily integrated in service design and automatically orchestrated, but this approach comes with several limitations in NFV-based services. As a matter of fact, programmers and service developers are not usually security experts, since security is usually managed by operation staff. In addition, deploying specific security appliances for each VNF duplicates operations, slows down the execution, and has a limited context knowledge that hinders the detection of complex multi-vector attacks. Integrating security appliances in graph design may lead to weak or ineffective protection, giving false trust confidence to service users. Instead, creating smarter orchestrators that can automatically embed the security in the service graph, possibly by enriching network functions with ad-hoc security services, opens a new set of opportunities.

Enable infrastructure-level trust and security. Virtualization in NFV is expected to adopt existing technologies from cloud computing (e.g., *OpenStack*), so it is natural to look for security architectures in that field. In this context, several factors, like the need for lightweight processing, the presence of a (untrusted) hypervisor layer, and multi-tenancy are wisely suggesting to pursue more distributed forms of monitoring and control, tightly integrated into the virtualization platform [15]. This approach is already intrinsic in the concept of *distributed firewall*, already available in many cloud management software, which integrates packet inspection and filtering in hypervisors. Distributed firewalls deploy packet inspection rules in hypervisors, enabling very fine-grained control over security policies, even beyond mere IP-based structure (through the notion of logical "containers" or "security groups"). However, a major limitation of distributed firewalls is that they cannot provide the same guarantees of private enterprise networks. Basically, they are embedded in the infrastructure, where trust mechanisms are still missing. Hypervisors and overlay networks provide isolation, but are not immune to attacks. DoS attacks against the physical network affect all virtual networks of all tenants, while a compromised hypervisor is a potential source of eavesdropping and alteration for every hosted virtual machine or software container.

Avoid security as a vertical silo. Finally, most security appliances are vertical silos. The lack of open cross-vendor public APIs requires an additional layer made of Security Information and Event Management (SIEM) software, which is not trivial to design and operate in a heterogeneous environment. Yet the nature of the NFV service permits routing and steering of malicious or infected traffic towards scrubbing centres in a natural way, while avoiding complex and inefficient solutions based on BGP or forwarding agents.

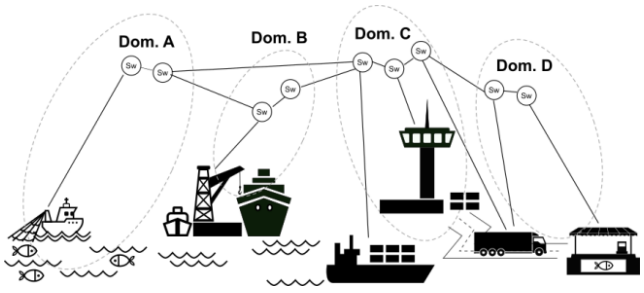

Fig. 2 Fish supply chain from the sea to the market.

3. A Use Case for 5G: Trusted Food Supply Chain

Food supply is a typical example of business chain where suppliers, shippers, dealers, and even consumers need to share information for both commercial and logistic purposes as well as nutrition care.

Let us consider the fish supply chain pictorially outlined in Fig. 2. Bringing the fish from the sea to the table entails a number of complementary commercial activities: fishing, sea shipping, road transportation, wholesaling, and retailing.

Every business has its own information processes, to account for sales, purchases, billing, goods tracking, employers, finance, etc. All these processes are increasingly implemented as *cyber-physical* systems, where IT resources are directly connected to smart things scattered everywhere. This is possible in 5G, where pervasive and programmable networks interconnect many computing facilities, enabling massive deployment of multiple (micro)-services across the entire infrastructure (edge, transport, core, cloud). However, modern trade and business models also require tight integration between interdependent domains, in order to know quantity and quality of goods, to schedule delivery times, to certify the whole supply chain, to automate invoicing, and so on.

For instance, the type, quantity, and quality of caught fish may be early shared by fishing boats to choose the more remunerative customer (e.g., wholesalers or fish industries), to size the container(s) for shipping, schedule embarking and delivery times. Then, during transportation on ship the temperature of the container is controlled and adjusted depending on the condition of the fish at embarking time, weather, and estimated date of arrival at the destination port. Also, ship tracking allows a terrestrial transport company to schedule the transport of the fish to the marketplace. Similar operations may be in place during transport on truck. Finally, the above interactions between different domains may rapidly vary over time, e.g., when (for a business reason) the fishing boat gets directly in touch with the terrestrial carrier. This prevents the establishment of manually operated trusted relationships between parties; instead, it requires the automatic and dynamic setup of security relationships between domains upon request.

The deployment of cyber-physical information systems is a complex process, which encompasses provisioning of IT resources, deployment, configuration and interconnection of software and smart things, definition of business relationships and enforcement of security policies. This also creates a security interdependence, because one compromised system becomes a potential security breach for any other intertwined domain.

Moving existing applications and services to the cloud brings more autonomy, through orchestration of software functions; further, the 5G concept promises new paradigms for cyber-physical systems, well beyond legacy middleware (e.g., FIWARE[†]). This transition also paves the way for new security models, which are able to guarantee trustworthiness among the different parties (users, 5G operators, cloud providers) and to effectively detect even complex and multi-vector attacks over multiple intertwined domains. For instance, the security infrastructure may permit an external enterprise to "see" only a subset of data available from one sensor, or to access only a subset of functionality of a business service. On the other hand, we need fine-grained security monitoring on data packets and application logs, to quickly and effectively detect and confine even multi-vector attacks that exploit vulnerabilities and breaches in different services/domains.

4. Concept and Approach

Virtualization provides elastic and cost-effective execution environments, but effective implementation, maintenance, and re-usage of complex business services also seek more automation in software deployment and lifecycle management. In this respect, a transition from "prescriptive" (i.e., procedural languages) to descriptive models is already ongoing, both for cloud applications [12], [13] and network function virtualization [14]. Such models describe the application as a logical topology of virtual functions; in addition to the business logic (i.e., the software), virtual functions also include metadata intended for automatic deployment and orchestration tools.

Metadata typically includes the name of the component (i.e., trademark and vendor), its description (including licensing and usage terms), provided functionality (e.g., EPC[††], eNodeB[†††], RAS[††††]), required services (e.g., database, authentication server), deployment constraints (e.g., number of cores, CPU speed, RAM, disk space, network bandwidth, hardware acceleration), measured performance metrics (e.g., packet latency and throughput, dropped packets, packet statistics), and management hooks (for instance, to start, stop, reload, or reset the service, to collect measurements, data, events, log). This information

[†]https://www.fiware.org/
[††]Evolved Packet Core.
[†††]eNodeB, or Evolved Node B, is the only mandatory node in the radio access network (RAN) of LTE. The eNodeB is a complex base station that handles radio communications with multiple devices in the cell and carries out radio resource management and handover decisions.
[††††]Radio Access and Spectrum.

is used by orchestration tools to provision the proper set of resources, set up and configure the execution environment, and perform life-cycle management actions (e.g., scale the application according to current traffic, react to failures).

New medadata and policies, specific for security aspects, can be introduced in the service definition, specifying high-level *intents* that need to be translated into the instantiation of proper security functions and the companion configuration rules. In fact, service models and orchestration are powerful paradigms that can boost effective and efficient management of security aspects, beyond the limitations already discussed in Sect. 2. In this respect, security should only be defined in terms of high-level policies and requirements at design stage. Instead, the selection and instantiation of security components may be automated at deployment time, and the current situation should be presented to cyber-security experts for triggering automated or manual reaction (see Fig. 3).

Security properties defines specific requirements and policies that affect the deployment and life-cycle management of the service graph. Examples of security properties that should be available include:

- the deployment of *cyber-attack detection* frameworks, which monitor the execution of the service graph and identify anomalies and other suspicious conditions;
- the deployment of additional components for *identity management and control access*, to enable seamless and secure interconnection with external components;
- the definition of *reaction policies*, in terms of actions to be executed when specific events occur or conditions are met, for instance according to a typical 'if-then-else' scheme;
- the inclusion of specific agents or independent functions to segregate or obfuscate data and traffic for *privacy* issues;
- the inclusion of specific agents or independent functions to intercept of traffic or retain data for *legal investigation and forensics*.

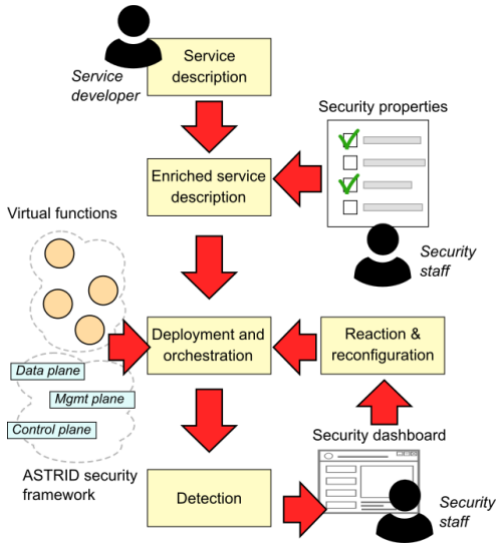

Fig. 3 Enhanced workflow for situational awareness in virtual services.

Figure 4 shows an example of service graph enriched with detection and identity management capability.

The deployment and orchestration process will then select the proper set of virtual functions that fit graph requirements and security constraints. The dynamic composition of software components needs descriptive metadata for each virtual function. In addition to the generic properties already described, security requirements should be present to describe the following capabilities:

- *security logs and events* generated by the virtual function;
- *encryption algorithms* to protect both user data as well as control data exchanged within the functions participating to the security framework;
- *trust and privacy requirements* for connecting and exchanging data with other functions;
- *identity and access control interfaces*, that can be used to authenticate external entities, set access rights, trace commands, etc.;
- *certification, timestamping, digital signature* capability that may be used to guarantee the origin and integrity of security data generated by the function;
- *traffic mirroring* and other replication capabilities that can be used for legal interception of data flows;
- *retained data* that may be used for criminal investigation and forensic tasks.

During deployment, orchestration takes care of in-

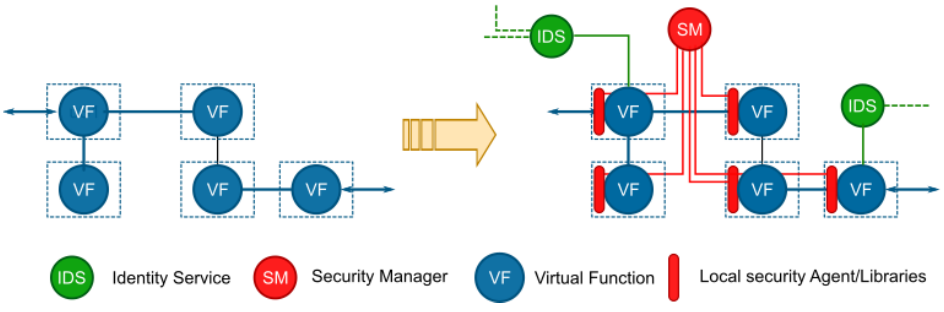

Fig. 4 Example of service graph enrichment with detection capability and identity management.

stalling all the libraries, proxies, and agents required by the security properties. The whole framework is responsible of continuous monitoring and inspection of the functions in the graph, collecting events and measurements, making them available to attack detection and identification of vulnerabilities and new threats, and enforcement of security policies (packet filtering, topology changes, etc.).

Security policies are the best way to respond to well-known threats, for which there are already established practice and consolidated methodologies for mitigation or protection. However, the identification of new threats and the elaboration of novel countermeasures requires direct step-by-step control over the on-going system behaviour.

Finally, the security dashboard is the main management tool for reporting indications about detected attacks and anomalies, to set run time security policies, and to perform manual reaction. In fact, although users may not be interested in specifying finely-grained security rules, they still need to be informed about the current state of their system. The dashboard interacts with the orchestration system to give security manager back full control over the graph in case of need.

5. The ASTRID Framework for Detection and Mitigation

One of the most challenging issue for implementation of the novel concept outlined in Sect. 4 is the definition of the framework for threat identification and attack detection. Indeed, according to our main objectives, we don't want to rely anymore on mere virtualization of existing security middleboxes; rather, we want to capture in a more integrated way data from heterogeneous sources, and to link them to the graph topology.

Figure 5 shows the layered architecture we are developing in the context of the EU project ASTRID[†]. It is based on the logical separation between three different planes, resembling the typical organization of communication infrastructures (though not directly related to network operations) and revolving around service orchestration. Instead of overloading the execution environment with complex and sophisticated threat detection capabilities, efficient processing capabilities are provided in the execution environment that create events and knowledge (*data plane*); algorithms for detection of threats and vulnerabilities are moved upwards and process such data in a coordinated way for the whole execution environment (*control plane*). The security framework is also able to dynamically re-configure the data plane, through specific policies that are interpreted by service orchestration (*management plane*).

5.1 The Data Plane

Figure 6 shows a more detailed view of the ASTRID data plane.

[†]https://www.astrid-project.eu/

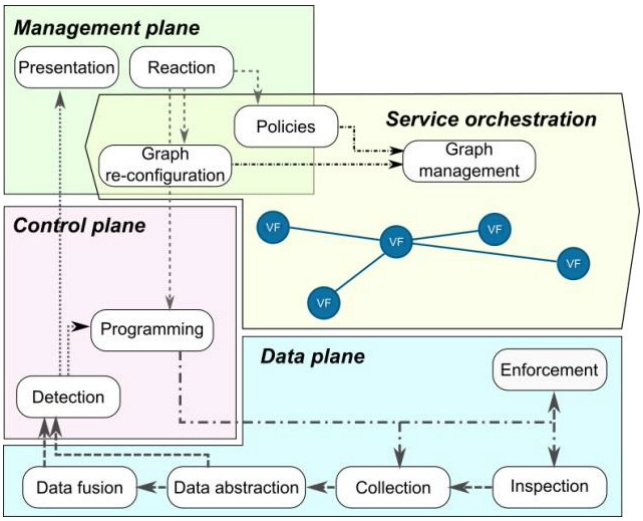

Fig. 5 The ASTRID multilayer architecture.

At the bottom of the architecture, the data plane concerns inspection of security-related data and information, their collection and aggregation into suitable abstraction, and data fusion techniques to correlate data and events from multiple sources. In addition, enforcement mechanisms are also present to filter packets, allow or deny execution of specific instructions, and so on.

The Data Plane is represented by multiple programmable security hooks, which are present in the virtualization environment. The hooks include logging and event reporting capability developed by programmers into their software, as well as monitoring frameworks built in the kernel and system libraries that inspect network traffic and system calls. Simpler hooks may limit to data reporting, but many of them should also include processing capabilities, to reduce the amount of network traffic generated. Security hooks are 'programmable' because they can be configured at run-time, hence shaping the system behaviour according to the evolving context. This means that packet filters, types and frequency of event reporting, and verbosity of logging are selectively and locally adjusted to retrieve the exact amount of knowledge, without overwhelming the whole system with unnecessary information. The purpose is to get more details for critical or vulnerable components when anomalies are detected that may indicate an attack, or when a warning is issued by cyber-security teams about new threats and vulnerabilities just discovered. This approach allows lightweight operation with low overhead when the risk is low, even with parallel discovery and mitigation, while switching to deeper inspection and larger event correlation in case of anomalies and suspicious activities, hence, being able to properly scale with the system complexity, even for the largest services (e.g., carriers large scale virtual networks, and worldwide mass applications as social nets).

Each node in the service graph hosts a Local Security Agent (LSA), in addition to the virtual function and the execution environment (virtual container or virtual machine).

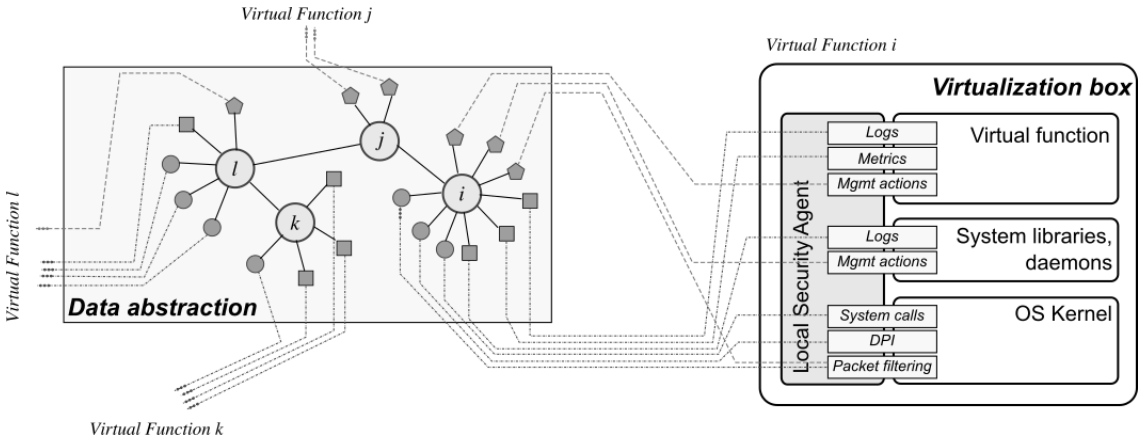

Fig. 6 The ASTRID data plane.

The LSA collects measures, events, and logs from the virtual function and the execution environment (system libraries, daemons, operating system kernel). Such information is then collected centrally.

The base for data abstraction is the topology of the deployed service. In this abstraction, each node represents a virtual function and each link a communication path. Satellites of nodes are security elements; they include both data plane capabilities (what can be collected, measured, and retrieved) and data (metrics, events, logs). Similarly, links have also properties (though not explicitly shown in the picture), related to the usage of encryption mechanisms and utilization metrics. In this abstraction, the overall topology and security capabilities are set by the orchestrator, whereas security data are fed by LSAs. The abstraction provides both real-time and historical information, hence allowing both on-line and off-line analyses.

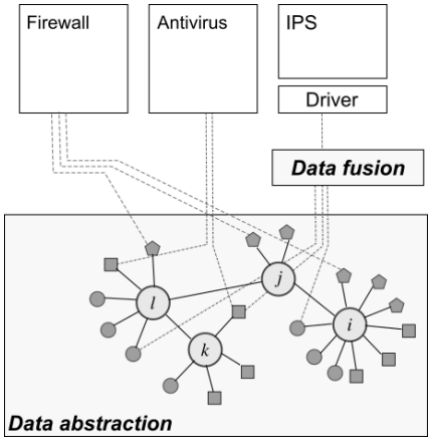

Fig. 7 The ASTRID control plane.

5.2 The Control Plane

Data abstraction in the data plane decouples the detection logic from the distributed data plane. Through the data plane, a common language can be used to query security-related attributes and to re-program inspection and enforcement tasks, without the need to use different interfaces and heterogeneous semantics.

The control plane clusters together typical functions currently available as separate appliances: Intrusion Prevention/Detection Systems (IPS/IDS), Network Access Control (NAC), Antivirus, Application Level Gateways (ALG), and more. One of the main advantages is the availability of data from different subsystems (disk, network, memory, I/O), instead of relying on a single source of information (network traffic) as is the common practice nowadays. The ambition is to effectively support identification of both software vulnerabilities and network threats, hence, involving a mix of source and run-time code analysis, formal verification, network analytics, and packet filtering techniques. This will assemble a diverse array of vulnerability analysis techniques to facilitate the transition of the application development industry to new security paradigms.

Through data abstraction, each security algorithm may access the data/control it needs (e.g., number of packets intended to given port on specific host, number of login failures, username used for failed authentication, etc.) for all virtual functions, hence building a global view on the overall system. The ambition is also to provide data fusion capabilities, so that pre-processing and aggregation of data may be accomplished by the same query, so to optimize look ups in the abstraction model.

As schematically depicted in Fig. 7, security algorithms may directly access the data abstraction, or relying on an ASTRID driver. This second option enables integration of legacy appliances into the system, by emulating their existing interfaces to access data.

The control plane is not a mere collection of detection algorithms. Since centralization of processing may easily result in excessive network overhead for collecting data and measures, it is important to shape the inspection, monitoring, and collection processes to the actual need. As described in the previous Section, the data abstraction also includes capabilities that are used for re-configuration of individual components and programming of their virtualization environ-

ments. In addition, these can be used to change the reporting behaviour, including parameters that are characteristics of each app (logs, events), network traffic, system calls (e.g., disk read/write, memory allocation/deallocation), RPC toward remote applications (e.g., remote DB). Programming also include the capability to offload lightweight aggregation and processing tasks to each virtual environment, hence reducing bandwidth requirements and latency.

Beyond the mere (re-)implementation of legacy appliances for performance and efficiency matters, the specific structure of the ASTRID framework paves the way for a new generation of detection intelligence, arguably by combining detection methodologies (rules-based, machine learning) with big data techniques; the purpose is to locate vulnerabilities in the graph and its components, to identify possible threats, and to timely detect on-going attacks. The combined analysis of security logs, events, and network traffic from multiple intertwined domains can greatly enhance the detection capability, especially in case of large multi-vector attacks. The challenge is clearly merging knowledge without exposing sensitive information to external domains. In this respect, the notion of local processing and distributed security analysis may provide an effective mechanism for multi-layer detection mechanisms, also exploiting artificial intelligence for identification of complex and unknown relationships between the domains.

The control plane basically corresponds to the Security Manager depicted in Fig. 4, so it looks like we are anyway inserting additional virtual functions in the service graph. Indeed, this component might be shared between multiple services, and this perhaps represents the best choice for security and efficiency reasons. As a matter of fact, the same detection algorithm may combine and correlate contextual information from all of them, which further improves the ability to react before a compromised system affects other services in a common business chain.

5.3 The Management Plane

The management plane includes high-level administration functions that interact with both human operators and software orchestrators. The main tasks are the representation and usage of situational awareness built by underlying security applications. The human interface is the interactive tool to draw the current cyber-security picture and to enable quick and intuitive response to attacks. It provides intuitive and easily understandable situational awareness to effectively support the decision process, by proper representation of the risk of possible attacks and the identification of threats and weaknesses (also including origin, positioning, dangerousness, replicability, etc.), and by enabling definition of custom reaction strategies in case of new and unknown threats.

Specific challenges include data and method visualization (e.g., to pinpoint the actual position of attacks and threats in the network topology, to point out the possible correlation between events in different domains), and decision support (e.g., to suggest remediation and countermeasures, to define automatic response to well-known attacks). Also, the presentation layer should provide seamless integration with CERT[†] networks to share information about new threats and attacks among different administrative domains (e.g., with STIX[††], in order to facilitate continuous update of the attack data base and the elaboration of common reaction and mitigation strategies [16]. Integration with existing risk assessment and management tools is also possible, so to automate most procedures that are currently still carried out manually. This will ultimately speed up the sharing and learning process, reducing reaction times and improving the overall resistance and resilience.

For reaction, two complementary options are available, as already discussed: direct human intervention or predefined security policies. In both cases, the reaction may affect the following objects.

- Configuration of the data plane: by setting enforcement policies at the network or software level (e.g., filtering packets, changing keys and other cryptographic materials, re-programming data collection for deeper inspection, etc.);
- Service graph: through life-cycle operations for replacing compromised functions with clean and more robust versions, upgrading the software, diverting plain network flows for legal interception, changing usernames/password and other configuration parameters, etc.

The definition of the management plane is strictly related to the choice of the orchestration tool. For instance, the graphical user interface may be separated or integrated with the dashboard for service deployment and management. In addition, there is not a common approach to define security policies (i.e., management operations to change the service graph).

6. Implementation

The implementation of the ASTRID framework is rather challenging, but several tools are already available and evolving in the envisioned direction.

For orchestration, OpenBaton [17] is currently among the most mature and flexible NFV solutions, and it is compliant with the ETSI MANO [18] framework. It supports design, deployment, and life-cycle management of data-driven service graphs, and suitable extensions have already covered service function chaining. Service graph enrichment is still unsupported, though, but this feature should require a limited effort.

Coming to the data plane, some frameworks are already available to collect logs from multiple sources, transport them to the centralized repository, and ensure data are not

[†]A computer emergency response team (CERT) is an expert group that handles computer security incidents.

[††]Structured Threat Information Expression (STIX) is a structured language for describing cyber threat information so it can be shared, stored, and analyzed in a consistent manner.

lost. One solution may be to configure log files to be monitored continuously, or relying on logging utilities such as *syslog*. Another approach is the definition of specific APIs for logging, so each application directly writes its log to the collection framework. Existing solutions like LogStash, Apache Flume, and Fluentd are capable of collecting from multiple sources, often use multi-tier architectures with fan-in topologies, support both polling and event-driven mechanisms, and can manages batches of events.

For network packets and events, eBPF[†] is currently the most advanced and flexible packet filtering tools available. eBPF enables arbitrary code to be dynamically injected and executed in the Linux kernel while at the same time providing hard safety guarantees in order to preserve the integrity of the system. While originally conceived to filter network packets only, it has now evolved to catch a broader set of kernel events; in general, any kernel event can be potentially intercepted (Kprobes, Uprobes, syscalls, tracepoints), making eBPF capable of analysing message (socket-layer) received, data written to disk, page fault in memory, files in /etc folder being modified. Recent projects proposed its usage also for the creation of complex network functions. The eBPF is a generic in-kernel, event-based virtual CPU, which leverages an assembly-like syntax for very efficient and quick processing. eBPF programs can be dynamically created and injected in the kernel at run-time. Assembly BPF bytecode is either interpreted or (in recent kernels) translated into native assembly code (e.g., x64) at run-time with a Just-in-time translator (JIT). Though running in kernel space, eBPF is safe because it is executed in a sandbox that prevents possible critical conditions at run-time; furthermore, a verifier checks the code and can refuse to inject it in the sandbox. eBPF runtime consumes a little amount of resources, so it cannot be used to generate a possible "denial of service" attack in the kernel because of its limited resource consumption. These are just the most prominent features that make eBPF as the perfect technology for inspecting network traffic and system calls in the ASTRID framework, overcoming the current limitations of the technologies as far as complex services are concerned [19].

On the abstraction side, the Neo4j database is suitable to store information and data with graph-oriented syntax. It has been already used to collect security events. Other possible solutions are OrientDB, ArangoDB, and ElasticSearch. Some of these tools can be queried by the GraphQL language, which can perform complex look-ups in a very efficient way, hence acting as a simple data fusion tool.

The control plane is perhaps the most challenging part of the framework, yet it is expected to bring the most relevant scientific advances. Existing algorithms already make use of flow-level information for network volume anomaly detection, though this only represents the crumbs of what may be available tomorrow. The availability of large datasets collected and related to whole service graphs opens a broad range of opportunities to successfully apply artificial intelligence and machine learning techniques not only for detection of attacks, but also for identification of new threats. In particular, new algorithms for vulnerability analysis and threat detection may be based on the ideas of the Attack Graphs, Attack Surface analysis, Kill Chain definitions and Attack trees models with the support of the deep learning techniques, petri nets, and game theory models. Correlation should also include automatic selection of the algorithms for the analysis of the threats based on the threat potential negative impact, both environment-dependent and environment-independent.

[†]Extended Berkeley Packet Filter. Overview available online at https://lwn.net/Articles/740157/.

7. Conclusion

Trustworthiness will be a major requirement for composing business chains over 5G infrastructures. Although manipulation of the service graph is expected to be an essential feature for some specific aspects (e.g., replacement of compromised software, legal interception, connection to security frameworks), deep and pervasive control over the execution environment remains the challenging issue to build wide situational awareness and timely react to attacks.

In this respect, the ASTRID framework represents a novel and still unexplored approach to improve current practice towards more efficiency and effectiveness. We are currently developing the lower layer (i.e., data plane), which is expected to boost soon novel algorithms for detection of known attacks and identification of new threats. The timeline of the ASTRID project is rather strict, with the full ASTRID architecture expected to be released at the beginning of 2019 and the software framework planned for the end of the same year. This schedule would enable potential early adopters to practice with the developed technologies in the near term, hence potentially provide hints for possible improvements in the coming months.

Acknowledgments

This work was supported in part by the European Commission, under Grant Agreement no. 786922 (ASTRID Project).

References

[1] D. Soldani and A. Manzalini, "On the 5G operating system for a true digital society," IEEE Veh. Technol. Mag., vol.10, no.1, pp.32–42, March 2015. DOI: 10.1109/MVT.2014.2380581

[2] I. Cerrato, A. Palesandro, F. Risso, M. Sune, V. Vercellone, and H. Woesner, "Toward dynamic virtualized network services in telecom operator networks," Elsevier Computer Networks, vol.92, pp.380–395, 2015. DOI: 10.1016/j.comnet.2015.09.028

[3] R. Bonafiglia, G. Castellano, I. Cerrato, and F. Risso, "End-to-end service orchestration across SDN and cloud computing domains," 2017 IEEE Conference on Network Softwarization (NetSoft), pp.1–6, Bologna, July 2017. DOI: 10.1109/NETSOFT.2017.8004234

[4] I. Cerrato, F. Risso, R. Bonafiglia, K. Pentikousis, G. Pongracz, and H. Woesner, "COMPOSER: A compact open-source service platform," Elsevier Computer Networks, vol.139, pp.151–174, 2018. DOI: 10.1016/j.comnet.2018.04.012

[5] "5G empowering vertical industries," whitepaper from the 5G-PPP, ERTICO, EFFRA, EUTC, NEM, CONTINUA and

Networld2020 ETP, https://5g-ppp.eu/wp-content/uploads/2016/02/BROCHURE_5PPP_BAT2_PL.pdf, accessed May 11th, 2018.

[6] A. Manzalini, D.R. Lopezz, H. Lonsethagen, L. Suciu, R. Bifulcozz, M.-P. Odinixi, G. Celozzixiii, B. Martinixv, F. Rissoy, J. Garayx, V. Foteinosk, P. Demestichasyy, G. Carullox, M. Tambascox, and G. Carrozzoxiv, "A unifying operating platform for 5G end-to-end and multi-layer orchestration," 2017 IEEE Conference on Network Softwarization (NetSoft), pp.1–5, Bologna, 2017. DOI: 10.1109/NETSOFT.2017.8004216

[7] R. Rapuzzi and M. Repetto, "Building situational awareness for network threats in fog/edge computing: Emerging paradigms beyond the security perimeter model," Future Generation Comp. Sys., vol.85, pp.235–249, Aug. 2018. DOI: 10.1016/j.future.2018.04.007

[8] G. Pék, L. Buttyán, and B. Bencsáth, "A survey of security issues in hardware virtualization," ACM Computing Surveys, vol.45, no.3, pp.40:2–40:34, 2013. DOI: 10.1145/2480741.2480757

[9] "Network Functions Virtualisation," whitepaper from the European Telecommunications Standards Institute, Oct. 2012. Available online at https://portal.etsi.org/nfv/nfv_white_paper.pdf

[10] ETSI NFV Industry Study Group, home page at https://www.etsi.org/technologies-clusters/technologies/nfv

[11] "Network Functions Virtualisation (NFV) Release 3; Security; System architecture specification for execution of sensitive NFV components," ETSI GS NFV-SEC 012 V3.1.1, Jan. 2017.

[12] B. Karakostas, "Towards autonomic cloud configuration and deployment environments," Intl. Conf. on Cloud and Autonomic Computing (ICCAC), pp.93–96, London, UK, Sept. 2014.

[13] J. Wettinger, U. Breitenbücher, and F. Leymann, "Standards-based DevOps automation and integration using TOSCA," IEEE/ACM 7th Intl. Conf. on Utility and Cloud Computing (UCC), pp.59–68, London, UK, Sept. 2014.

[14] P. Bellavista, L. Foschini, R. Venanzi, and G. Carella, "Extensible orchestration of elastic IP multimedia subsystem as a service using open baton," 5th IEEE Intl. Conf. on Mobile Cloud Comp., Services, and Engineering (MobileCloud), pp.88–95, San Francisco, California, USA, April 2017.

[15] M. Ali, S.U. Khan, and A.V. Vasilakos, "Security in cloud computing: Opportunities and challenges," Information Sciences, vol.305, no.1, pp.357–383, June 2015. DOI: 10.1016/j.ins.2015.01.025

[16] F. Skopik, G. Settanni, and R. Fiedler, "A problem shared is a problem halved: A survey on the dimensions of collective cyber defense through security information sharing," Elsevier Computers & Security Journal, vol.60, pp.154–176, July 2016. DOI: 10.1016/j.cose.2016.04.003

[17] A.M. Medhat, G.A. Carella, M. Pauls, M. Monachesi, M. Corici, and T. Magedanz, "Resilient orchestration of service functions chains in a NFV environment," 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp.7–12, Palo Alto, CA, 2016.

[18] ETSI GS NFV-MAN 001 v1.1.1, Network Functions Virtualisation (NFV) Management and Orchestration. Available online at https://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf

[19] S. Miano, M. Bertrone, F. Risso, M. Tumolo, and M. Vasquez Bernal, "Creating complex network services with eBPF: Experience and lessons learned," Proc. IEEE High Performance Switching and Routing (HPSR18), Bucarest, Romania, June 2018.

Stefan Covaci is senior solutions architect for next generation internet services platforms at the Computer Sciences and Electrical Engineering faculty of the Technical University of Berlin, Institute for Telecommunication Systems. He is also senior scientist in the Software Networks department of the Fraunhofer Institute FOKUS. His recent work is in the areas of 5G/next generation terrestrial and satellite networks infrastructures and service delivery platforms with a focus on programmability, interoperability, security and management solutions.

Matteo Repetto is Research Associate at CNIT, where he works in the National Lab of Smart, Sustainable, and Secure Internet Technologies and Infrastructures (S3ITI). His recent area of interest includes energy efficiency, virtualization and 5G infrastructures, network programmability and security.

Fulvio Risso received the Ph.D. in Computer Engineering in Computer Engineering in 2000 from Politecnico di Torino, Italy, where he is currently Associate Professor. His research interests focus on high-speed and flexible network processing, Software Defined Networks, Network Functions Virtualization. He has co-authored more than 100 scientific papers.