Using the COMPA Autonomous Architecture for Mobile Network Security

Liam Fallon
Ericsson
44 PUBLICATIONS **99** CITATIONS
SEE PROFILE

John Keeney
LM Ericsson Ireland
81 PUBLICATIONS **658** CITATIONS
SEE PROFILE

Sven van der Meer
L.M. Ericsson
116 PUBLICATIONS **965** CITATIONS
SEE PROFILE

Using the COMPA Autonomous Architecture for Mobile Network Security

Liam Fallon, John Keeney, Mark McFadden, John Quilty and Sven van der Meer
Network Management Lab, Ericsson, Athlone, Co. Westmeath, Ireland
{liam.fallon john.keeney mark.mcfadden john.quilty sven.van.der.meer}@ericsson.com

***Abstract*—The COMPA (Control, Orchestration, Management, Policy, and Analytics) adaptive control loop realizes an automation pattern that can operate recursively at many layers in a carrier network. An overall COMPA autonomic control loop can orchestrate functions, themselves implemented as COMPA autonomic loops. Thus the COMPA automation patter can recurse right down to resource level in a network. One of the most exciting application areas for the COMPA automation pattern is in assuring mobile network security. The recursive nature of the pattern is the ideal mechanism for automating monitoring and root cause analysis of security threats to networks.**

In this paper we present a Proof of Concept of a COMPA compliant system for a Distributed Denial of Service (DDoS) scenario. The system monitors, performs root cause analysis, and mitigates a DDoS attack. The system was built by integrating a number of existing components that were deployed as VNFs. Our experiences of using the system were that the system could handle a DDoS attack quickly and automatically. In addition, the system was very flexible to build and deploy.

***Index Terms*—COMPA, Autonomic Management, Mobile Network Security, Virtualization, NFV, VNF**

I. Introduction

To quote Victor Hugo ”Nothing is more powerful than an idea whose time has come”. This is certainly true of Autonomic Networking, which has a long research history [1] but, until recently, has only seen limited adaptation in the wild. However, with the advent of the *soft networking* that Software Defined Networking (SDN) and Network Function Virtualization (NFV) [2] enables, there is a renewed interest in Autonomic Networking, as it is seen as having the flexibility and inherent automation that is required to manage the complex and transient networks that are possible with these new technologies. It is perhaps not surprising therefore that we now see the emergence of autonomic networking architectures such as COMPA from network vendors [3] and ECOMP from network operators [4].

Mobile network security would seem to be a natural domain of application for a system using a COMPA autonomic control loop. Mobile network security use cases call for a system to monitor the network at a high level, identify possible threats, perform root cause analysis on those threats, and, if possible, mitigate those threats. Such use cases can be implemented as a loop that iterates towards a solution as more information on a threat is solicited from the network.

In the work described in this paper, we examined just such a use case. In order to assess how successful a system developed using a COMPA autonomic control loop would be in addressing a Distributed Denial of Service attack use case. We developed a COMPA compliant Proof of Concept system by integrating existing virtualized components. We deployed the system in our test lab and in the lab of a North American network operator. We demonstrated that the system could quickly detect, identify, and mitigate DDoS attacks and that the system could be entirely virtualized.

This paper is structured as follows. We present COMPA in §II. In §III, we describe how COMPA can be used to implement mobile security use cases. We describe how we applied COMPA to a mobile network DDoS attack scenario in §IV. In §V we describe how we implemented a Proof of Concept for the DDoS scenario, and in §VI we describe our experiences in using the COMPA approach and in building and running the Proof of Concept system.

II. COMPA

COMPA is derived from the key tasks of *Control*, *Orchestration*, *Management*, *Policy*, and *Analytics*. It provides for a modular framework and subsequently an architecture for a federation of control units exhibiting a common underlying control loop pattern. This underlying pattern is the COMPA adaptive control loop, effectively realizing an automation pattern. The complete target architecture is described in [3].

COMPA operates at different levels in a carrier network. At a given layer, it orchestrates and manages the layers resources to deliver services. Those resources may in turn contribute to the services exposed by the receiving layer. This realizes a recursive pattern with multiple, nested levels of control. The underlying building blocks of COMPA are: Control/Orchestration/Management (*COM*) executing control decisions; Policy (*P*) responsible for intelligence and dynamic governance to achieve control decisions; and Analytics (*A*) providing real time insights to influence control decisions.

The COMPA architecture defines a number of sample federated domain layers (responsibility domains) with specific functionalities shown in Figure 1: “Customer/Tenant” (service and cloud user), “Network Functions” (RAN, fixed access,

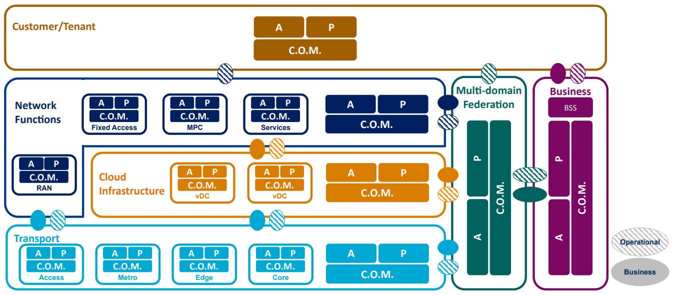


Fig. 1. COMPA - Architecture & Responsibility Domains

MPC), “Cloud Infrastructure” (Data Centers), and “Transport Networks” (Access, Metro, Edge, Core). These operational layers are supported by the cross-cutting “Multi-Layer Federation” and “Business” layers for continuous operations.

Within each domain, inside each adaptive component we find then the *C*, *O*, *M*, *P*, and *A* components that manifest an adaptive, closed control loop as the main automation pattern of COMPA. Conceptually the overall COMPA framework is a simple replication of a control loop as multiple nested instances. When implemented this pattern allows for multi-dimensional configuration and customization to be adapted to the requirements of the domain/layer it resides in.

The loop is based on long-standing work in autonomic networking [1]. Figure 2 shows three of the four different flows (top level interactions are not shown here) to manage an underlying automation target. Translated into the COMPA framework, the automation target being managed are those services exposed by a lower layer.

The red loop begins with Analytics (*A*) to lift, analyze, and process the incoming data (event streams from the automation target and context sources). Here ‘lift’ means to normalize, mediate, and correlate the data by transforming streams into a normalized form that all other parts of the control loop can understand. ‘Analyze’ refers to use of machine learning and statistical analytics to discover and understand trends and patterns in the event stream. ‘Process’ means to monitor events and match understood patterns. The results of this component are patterns and predictions as described in [5][6][7].

Patterns and predictions are then passed to the Policy (*P*) step. This component can recommend actions (recommendations, see [8]) or automatically decide on actions (request). Results here are recommendations and/or action requests.

COM performs semantic, functional and non-functional validation on recommendations and requests. For example, a semantic evaluation would test the action against a semantic model to infer any potential deviations, and functional validation would test if the action is functionally valid. Non-functional validation would then verify non-functional concerns such as security/access control rights, resource availability, affected services, etc. The validation process includes the final selection of an action (or a set of actions) that realize the effect required by the original recommendation or request.

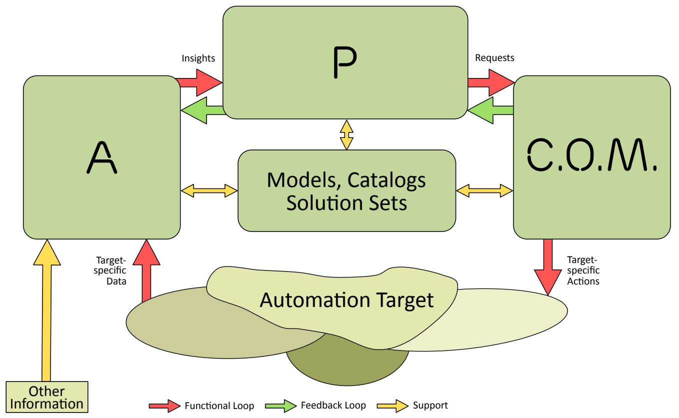


Fig. 2. COMPA - Autonomic Control Loop

Once validated, the action(s) can then be applied by translating the technology-neutral action(s) specification into the vendor-, node-, domain-specific representations required to enforce and commit the actions(s).

The green internal feedback flow supports the loop to self-stabilize, i.e. to reach equilibrium. Feedback from validations can therefore be used to direct the decision making. Feedback from the decision making about input patterns and predictions and user feedback can also be used to tune analytics and complex event processing.

Since the loop represents a dynamic system, we need a static set of functions that facilitate the dynamic aspects of the loop while enabling a consistent and stable operation. This supporting flow is represented by the yellow arrows to and from the “Models, Catalogs, Solution Sets” component. This component supports all other components with models of the artefacts and semantics, catalogs of patterns, context, algorithms and solution sets. Without this supporting flow the loop cannot operate consistently.

Details on how the COMPA architecture and the adaptive control loop realizes multi-domain, adaptive automation in the current and future business environment are discussed in [9]. An example of how the control loop is used to build a predictive, recommendation based network management system is presented in [6].

III. Using COMPA for Mobile Network Security

One of the goals of network operation and management is to ensure that subscribers can access network services in a fair and secure manner according to their subscription. While equipment reliability and careful network and service configuration and optimization are key enablers to achieve this goal, one major threat surrounds malicious (or accidental) misuse of the network in a way that might affect the network and the services that depend on the network. Therefore it is necessary to be constantly vigilant to detect and mitigate such misuse.

Some other goals of mobile network security include: to protect of network resources; to protect private and enterprise customer satisfaction and trust; to control operational costs from over-provisioning; to minimize time and expense in mitigating and recovering from network misuse; and to create value added services for subscribers, such as protecting them from remote attacks or block any of their resources that might host an attack. As with most network and service management tasks the goal is to increase automation and intelligence in mobile security tasks.

The COMPA control loop described above is directly applicable for both reactive and proactive mobile network security management. This approach can be used to detect occurrences of resource misuse or cyber security attacks and respond with corrective or mitigating actions to ensure continuous protection of critical network infrastructure. New and existing network and service monitoring tool can be used to collect information and analyze network and service performance with the goal to identify security-related anomalies in the network. Such anomalies can be further tracked, analyzed and classified as either unpredicted transient but acceptable resource usages, or alternatively as varying degrees of accidental or malicious threats to the network. Classification of anomalies can be performed by correlating the anomalies against predefined or learned thresholds or alternatively using threat intelligence gathered over time. Policies can also exploit additional context and decision making intelligence to further classify anomalies and to make decisions on how specific anomalies or threats should be handled when they occur. Existing approaches for network and service control, orchestration and management can then be used to enforce those decisions in a consistent manner. COMPA control loops then continue to monitor, analyze, make decisions and enforce decisions on the managed network in a way that continues to provide appropriate service for subscribers.

In summary, a COMPA-based approach for mobile network security would follow the following general steps:

1) Detect potentially unusual behavior, perhaps by monitoring network or service KPIs or events.
2) Analyze and classify the unusual behavior, perhaps requesting more detailed service information. Known and acceptable service behaviors may be logged and then removed from further consideration.
3) Closer in-depth and specialized observation of suspect services to further determine whether the behavior is harmful or acceptable. Again behaviors that are not harmful can be removed from consideration.
4) Service traffic flows that are determined to be harmful or remain unknown can be dealt with using appropriate policies. Some candidate responses might include: increased logging for the flow; block, redirect or quarantine the flow; warn the subscriber that is responsible for or is the target of the flow; capture and further analyze the flow; etc..
5) When service behaviors return to normal, perhaps after some timeout, any specialized handling or filtering of flows should be disabled as appropriate.

IV. A Mobile Security COMPA Proof of Concept

We have built a proof of concept that demonstrates how an autonomic COMPA compliant system can be used to isolate and neutralize a distributed denial of service (DDoS) attack on a mobile network.

A. DDoS Scenario

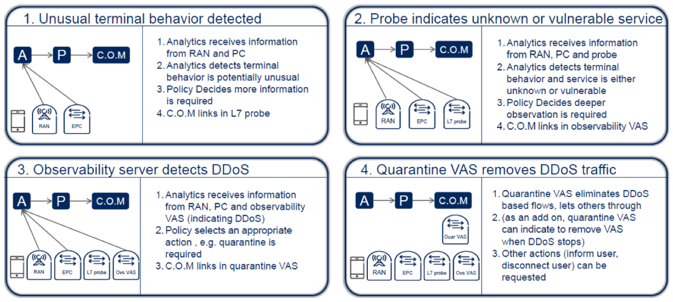
In our scenario we examine the case where a number of subscribers' resources (e.g. phones) have become infected with a virus that triggers a coordinated Distributed Denial of Service (DDOS) attack targeting some resources either inside or outside of the operator's network. The attack might manifest itself by creating a large number of data connections, which may in turn result in a large increase in control signaling on the mobile network. For example, a particular case might cause mobile phones to repeatedly attach themselves to and detach themselves from the network, thus potentially creating a signaling storm in the network. As another example, the virus might triggers bursts of user sessions to be established and torn down. This malicious behavior causes high signaling loads in the network and impedes the handling of legitimate signaling, thus potentially affecting traffic from other legitimate subscribers.

B. Toolbox Available

In a managed mobile network the Operations Support System (OSS) is constantly gathering and processing monitoring events and KPIs about the current state of the network, subscribers and the services they use.

- Example monitoring data streams include:
  - Event-based Monitoring (EBM) events generated at the SGSN/MME network control node. A event is generated to describe each time a subscriber attaches or detaches from the network and each time a network service request or traffic bearer is established or ends.
  - Cell trace (CTR) events can be enabled to describe the low-level performance of individual radio network cells. For example CTR events contain low-level KPIs describing the radio characteristics of the cells, traffic volumes, mobility events, etc..
  - User Equipment trace (UEtrace) events can be enabled to describe the low-level performance of individual subscribers' connected equipment (UE) (e.g. phone or modem). For example UEtrace events contain low-level KPIs describing the radio characteristics of the UE, traffic volumes, mobility events, etc..
  - User-plane Deep Packet Inspection (DPI) events can provide summary information about individual user-plane traffic flows, for example the source/destination addresses and ports, an approximate classification of the type of traffic, the traffic volumes, etc.

  Note, EBM events, CTR events, UEtrace events and user-plane monitoring events use different identifiers to identify users, UEs, cells, traffic flows, bearers and endpoints. A significant amount of event mediation and correlation is required to correlate events from different sources and to generate consistent identifiers across sources, for example, using Cell Trace UE Mapping (CTUM) events generated by the MME in an LTE network.
- To process and correlate the various event streams the Ericsson xStream [10] platform is used. Stream Analytics functions written in a variety of languages (e.g. R, Esper, Java, etc) can be embedded within xStream to process event event flows.
- The Apex Policy Engine [11] is used to make decisions about abnormal traffic flows should be handled.
- In a mobile network the 3GPP Policy and Charging Rules Function (PCRF) is used to authorize and configure the user-specific QoS requirements for user-plane traffic flows. This function can be used to temporarily downgrade or disable traffic flows for a given subscriber. For ease of use it may be advantageous to use portions of an



1. Unusual terminal behavior detected

1. Analytics receives information from RAN and PC
2. Analytics detects terminal behavior is potentially unusual
3. Policy Decides more information is required
4. C.O.M links in L7 probe

2. Probe indicates unknown or vulnerable service

1. Analytics receives information from RAN, PC and probe
2. Analytics detects terminal behavior and service is either unknown or vulnerable
3. Policy Decides deeper observation is required
4. C.O.M links in observability VAS

3. Observability server detects DDoS

1. Analytics receives information from RAN, PC and observability VAS (indicating DDoS)
2. Policy selects an appropriate action , e.g. quarantine is required
3. C.O.M links in quarantine VAS

4. Quarantine VAS removes DDoS traffic

1. Quarantine VAS eliminates DDoS based flows, lets others through
2. (as an add on, quarantine VAS can indicate to remove VAS when DDoS stops)
3. Other actions (inform user, disconnect user) can be requested

Fig. 3. AADM Scenario

OSS system (e.g. Ericsson ENM [12]) to interface with the PCRF control node.

- User-plane Deep Packet Inspection probes can be used to observe and classify user-plane traffic for particular traffic flows [13].
- An Observability Server can used to perform deep inspection of small portions of user-plane data, for example to identify virus signatures in user traffic.
- A Quarantine Service can act as a terminal sink to log and discard undesired traffic, or alternatively to route undesired traffic through a quarantine network slice in a manner that does not affect other traffic in the network.
- A Virtual Function host environment (e.g Openstack or Ericsson Cloud Manager [14]) is required to host, start and stop each of the various virtual functions that make up the management network and its managements systems.
- An SDN controller can be used to mirror and route individual traffic flows to the different components of the system [15].

C. Applying COMPA

We applied the COMPA compliant generic approach described in §III to design and build our test system. Fig.3 shows how the proof of concept handled this DDoS scenario.

1) Detect unusual behavior: The Analytics component monitors control session information from base stations in the RAN (Radio Access Network) and from nodes in the PC (Packet Core) via COM. Analytics detects in real time that there is a spike in the amount of control signaling and that the control signaling is exhibiting unusual patterns. Analytics notes that certain users have made multiple connection attempts and generates an *infection severity index* that is weighted based on the type, number and frequency of attempts. The Analytics step then passes an event to the Policy component describing its observations and supplying a list of the potentially abnormal sessions.

Policy examines the observations from Analytics, particularly the *infection severity index*, and decides that more detailed information is required on some of the sessions that have been observed. Policy makes the decision that, whilst the signaling is unusual, the signaling may be legitimate and so requires more information before a more definitive decision can be made. It request the COM component to turn on Layer 7 probing for the particular sessions in question to get more detailed information on those sessions.

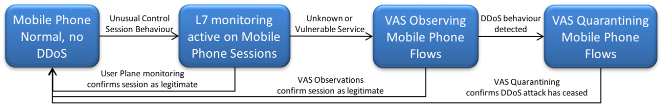

Mobile Phone Normal, no DDoS
Unusual Control Session Behaviour
L7 monitoring active on Mobile Phone Sessions
Unknown or Vulnerable Service
VAS Observing Mobile Phone Flows
DDoS behaviour detected
VAS Quarantining Mobile Phone Flows
User Plane monitoring confirms session as legitimate
VAS Observations confirm session as legitimate
VAS Quarantining confirms DDoS attack has ceased

Fig. 4. States in the DDoS Mitigation Policy

2) User plane probe information diagnoses root cause: Once probing has been enabled and the particular session continue to behave abnormally the Analytics component observes probing information in addition to the RAN and PC observations. For some sessions Analytics can now detect that the terminal user plane session is of an unknown type or that the signaling behavior does not match the expected characteristics for the user plane traffic observed for a session. The Analytics component again passes its observations and session list to the Policy component.

The Policy component may decide to immediately quarantine the traffic (selects action 3.3 in Fig.3), for example in the case of a session being of an unknown type. Alternatively, it may decide to enable an Observability Value Added Service (VAS) in the form of third party virus detection software if the signaling behavior does not match the expected characteristics for the user plane traffic observed for a session. Policy requests COM to deactivate the expensive user-plane probing on the session and to enforce the new action it has selected.

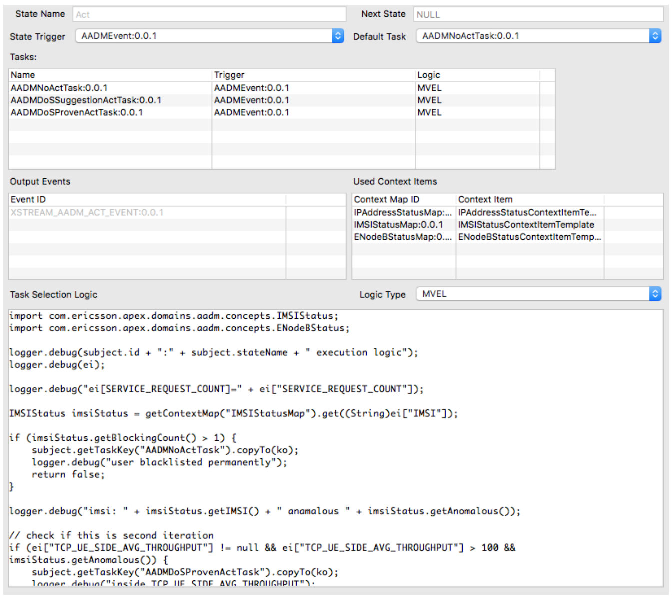
3) Observability VAS detects DDoS: Analytics now observes messages from the Observability VAS from the affected sessions as well as from the monitoring information from the RAN and PC. Analytics confirms that a DDoS has been detected and again passes its observations and session list on to the Policy component.

Policy now decides how to deal with the threat that has been identified. One of a number of mitigation strategies can be used. Policy may decide to use a quarantine VAS in the form of a special network slice to which quarantined traffic is redirected. It may decide to notify the Customer Relationship Management system so that user support staff can contact the end users or it may request an SMS or email be sent directly to the affected subscribers. Alternatively, Policy may decide to log the incident and take no action, for example in the case of VIP customers. Policy then request the COM system to deactivate the Observability VAS and to implement the new action it has selected.

4) Quarantine VAS removes DDoS Traffic: COM receives an action event describing the decision made by the Policy component and carries out the action selected by Policy.

D. The DDoS Mitigation Policy

In our scenario, the Policy component controls the autonomic behavior of the system and also holds state about relevant sessions, subscribers and devices as the scenario progresses for each subscriber. That state is shown in Fig.4. In our scenario, the COM component is stateless; it forwards events from the network and applies actions on the network required by Policy. In this scenario, the Analytics component


State Name: Act
Next State: NULL
State Trigger: AADMEvent:0.0.1
Default Task: AADMNoActTask:0.0.1

Tasks:

Name	Trigger	Logic
AADMNoActTask:0.0.1	AADMEvent:0.0.1	MVEL
AADMDoSSuggestionActTask:0.0.1	AADMEvent:0.0.1	MVEL
AADMDoSProvenActTask:0.0.1	AADMEvent:0.0.1	MVEL

Output Events

Event ID
XSTREAM_AADM_ACT_EVENT:0.0.1

Used Context Items

Context Map ID	Context Item
IPAddressStatusMap:...	IPAddressStatusContextItemTe...
IMSIStatusMap:0.0.1	IMSIStatusContextItemTemplate
ENodeBStatusMap:0....	ENodeBStatusContextItemTemp...

Task Selection Logic
Logic Type: MVEL

```
import com.ericsson.apex.domains.aadm.concepts.IMSIStatus;
import com.ericsson.apex.domains.aadm.concepts.ENodeBStatus;

logger.debug(subject.id + ":" + subject.stateName + " execution logic");
logger.debug(ei);

logger.debug("ei[SERVICE_REQUEST_COUNT]=" + ei["SERVICE_REQUEST_COUNT"]);

IMSIStatus imsiStatus = getContextMap("IMSIStatusMap").get((String)ei["IMSI"]);

if (imsiStatus.getBlockingCount() > 1) {
    subject.getTaskKey("AADMNoActTask").copyTo(ko);
    logger.debug("user blacklisted permanently");
    return false;
}

logger.debug("imsi: " + imsiStatus.getIMSI() + " anamalous " + imsiStatus.getAnomalous());

// check if this is second iteration
if (ei["TCP_UE_SIDE_AVG_THROUGHPUT"] != null && ei["TCP_UE_SIDE_AVG_THROUGHPUT"] > 100 &&
imsiStatus.getAnomalous()) {
    subject.getTaskKey("AADMDoSProvenActTask").copyTo(ko);
    logger.debug("inside TCP_UE_SIDE_AVG_THROUGHPUT");
```

Fig. 5. The AADM Policy in a Policy Editor

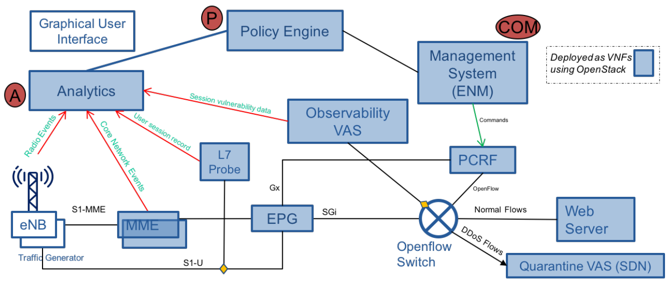


Fig. 6. Proof of Concept Testbed

is also stateless; while aggregates and correlates information coming from the network via COM, once it has finished its analytics tasks it forwards its results to Policy and discards its state. Since it is necessary for the Policy system to preserve state, it must use a mechanism such as shared memory or persistent storage to hold state for each relevant subscriber and terminal.

The DDoS mitigation policy was implemented as a Dynamically Adaptive Policy [16] in the Apex Policy Engine [11]. A screenshot of part of the policy is shown in Fig.5.

V. Proof of Concept Implementation

We built and deployed the COMPA proof of concept testbed in the Ericsson NM Lab and also in the lab of a North American network operator.

A. Testbed

The testbed layout is shown in Fig.6. A traffic simulator was used to simulate a LTE radio access network with five eNodeB nodes and 2,000 active subscribers. Malicious behavior was initiated on 10% of the subscribers.

A NFV [17] deployment was used for all other network elements (shaded in blue in Fig.6). The VNFs were deployed using OpenStack. The traffic flows pass from the eNodeBs in the traffic generator via the MME (Mobility Management

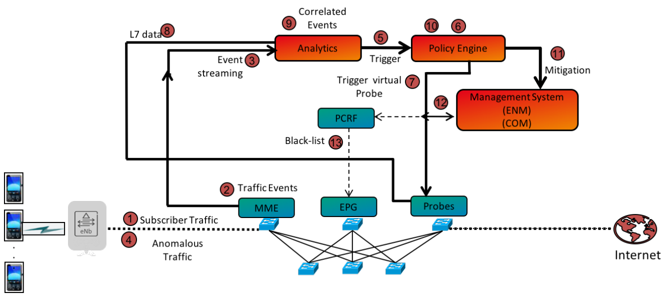


Fig. 7. Autonomous Flows in the Testbed

Entity) and the EPG (Evolved Packet Gateway) to an OpenFlow switch[1]. The Analytics component, the Policy Engine, and the ENM [12] management system make up the *A*, *P*, and *COM* parts of the COMPA autonomic loop. The action from the COMPA loop results in Openflow messages being sent to the Openflow switch, which determines the action taken on each flow in the generated traffic. The Openflow switch may duplicate flows to the Observability VAS, and will send flows onto the web server if a flow is determined to be normal or to the Quarantine VAS if the flow is determined to be DDoS traffic. Because the system was virtualized, the probes and Observability VAS were spun up on demand.

B. Autonomous Flows

The autonomous flows in the testbed are labeled in Fig.7 and the interactions at the labeled points are described in the list below:

1) Normal subscriber signaling traffic from multiple eNodeBs; signaling messages such as ‘Attach’s to and ‘Detach’s from the network and service requests.
2) The MME sends traffic events to Analytics when it sees messages such as Attach messages, Detach messages, and service requests for users
3) The Analytics component continuously receives and analyses the traffic events and
   - calculates average traffic per subscriber
   - counts the service requests
   - determines if a subscriber breaches a predefined threshold and, if so, the traffic event and subscriber statistics are used to generate an incident
4) Anomaly traffic simulation starts
5) Analytics detects the abnormality in the network and communicates the incident to the Policy Engine
6) The Policy Engine uses its logic and its saved state (see Fig. 4) to decide on the next action to be taken
7) In the case where user plane information is required, the Policy Engine sends a request to the probe to get additional information on the user session
8) The probe sends summaries of the Layer 7 user plane data to the Analytics component for further processing.
9) Analytics correlates the traffic events from the MME and the Layer 7 user plane information and sends the

[1]See [18] for a description of the 3GPP mobile network architecture and its network elements.

correlated result to the Policy engine. The correlated result includes a summary of the traffic events from the MME and information on the type of user plane application information detected on the user plane flow by the probe.

10) The Policy Engine uses its internal rules to decide if the correlated result is normal or DDoS traffic. If the traffic is not normal, it makes the decision to quarantine the malicious traffic
11) The Policy engine collects the subscriber identity and IP addresses of the terminal participating in the DDoS event and requests the management system to take action
12) The management system sends a command with the subscriber identity and IP addresses of the terminal participating in malicious traffic to the PCRF node.
13) The PCRF black-lists the malicious terminal for a configurable number of minutes (quarantine) while the control loop continues to monitor the traffic. If the DDoS persists, the Policy engine takes the protection level to next step and blocks the terminal indefinitely.

VI. Experiences and Summary

This trial demonstrated that inline event collection, correlation and analytics combined with real-time machine processing and policy execution was able to quickly and automatically deal with security incidents, and only the affected subscribers were impacted by the action of the control loop. The complex behavior needed to govern the use case could be handled via single Apex MEDA policy. The decoupling of Analytics, Control and Policy was important here. The same analytics event triggered the policy for each loop, the only difference each time is the amount of context contained in the event. The context-aware policy could then decide on different appropriate action requests towards the mobility controller, the policy was not aware of how the mobility controller acted on the action request. Each modular aspect of the control loop's operation was compartmentalized, thus simplifying the operational aspects of the control loop.

A potential drawback of this approach is the added complexity on the design side. The autonomic use case requires the coordinated creation of an analytics insight reports (CEP and/or ML rules), a policy (MEDA in this case), and controller workflows. These dependencies then need to be tracked as the goals and actions of loop might evolve over time. While Apex policies are more complex than traditional Event Condition Action (ECA), Condition Action (CA) or Goal policies, the policies themselves are much more expressive, adaptive and context-aware. This results in fewer self-contained policies, as seen in this example where only one policy was required. This allows for simpler Policy management (fewer self-contained policies, with less overlap between policies, simplifies conflict detection/mitigation, and it is easier to maintain and alter the use case behavior in the future).

The use of virtual network functions and systems greatly simplified the logistics involved in implementing this usecase. This allowed us to inject and remove different VNFs into the service chain in real time, for example the virtual probe (vProbe), the Quarantine VAS and the Observability VAS. Without the use of cloud-based virtual functions and flexible SDN-controlled networks such a scenario would have a high cost and require months of planning and deployment. Even if possible there would still remain issues with pre-provisioning all the necessary hardware and network paths for the use case (e.g. a hardware-based network tap for the network probe would need to be connected in-line in the user-plane traffic connections). Another important advantage of using an SDN-controlled network is that more flexible and dynamic routing capabilities and service chaining allows specific network flows to be re-routed and mirrored in a more flexible manner. This allows only specific flows to be redirected to the network probe, Quarantine VAS and the Observability VAS, and only when needed. This greatly reduces the resource requirements for these functions, while a cloud based implementation of these function allow more flexible resource allocation to these functions when required.

In summary, we have presented a prototype implementation of a COMPA control loop for a mobile network security use case around the detection and mitigation of a DDoS attacks within the network. Such an approach demonstrates how to improve situational awareness of illicit and spurious signaling and control traffic within a mobile network. This approach also demonstrates how the handling of such mobile security incidents can be handled in a more automated manner, while still supporting the flexibility that comes from continuous analytics and policy-based management. This work also demonstrates how customer experience can be improved with more proactive detection mitigation and warning of issues that would likely result in customer complaints related infected devices or service unavailability resulting from malicious traffic.

Acknowledgement

This work is partly funded by the European Commission through the ARCFIRE project (Grant 687871) under the H2020 program. The author also wish to thank the Customer Unit for the relevant operator for setting up and executing this trial.

References

[1] N. Agoulmine, *Autonomic Network Management Principles: from Concepts to Applications.* Elsevier Academic Press, 2010.

[2] ETSI, "Network functions virtualisation (NFV); Management and Orchestration," ETSI, Tech. Rep. GS NFV-MAN 001, v.1.1.1, 2014.

[3] G. Rune *et al.*, "Architecture Evolution for Automation and Network Programmability," *Ericsson Review*, no. 3, pp. 2–10, 2014. [Online]. Available: http://www.ericsson.com/res/thecompany/docs/publications/ericsson_review/2014/er-evolved-network-architecture.pdf

[4] AT&T, "ECOMP (Enhanced Control, Orchestration, Management & Policy) Architecture White Paper," AT&T, Tech. Rep., 2016. [Online]. Available: http://about.att.com/content/dam/snrdocs/ecomp.pdf

[5] F. Zaman *et al.*, "i-magnet: A real-time intelligent framework for finding specific needles from needle stacks," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, May 2015, pp. 846–849.

[6] ——, "A Recommender System Architecture for Predictive Telecom Network Management," *Communications Magazine, IEEE*, vol. 53, no. 1, pp. 286–293, 2015.

[7] S. Robitzsch *et al.*, “E-stream: Towards pattern centric network incident discovery and corrective action recommendation in telecommunication networks,” in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, May 2015, pp. 842–845.

[8] J. Keeney, S. van der Meer, and G. Hogan, “A Recommender-System for Telecommunications Network Management Actions,” in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*. Gent, Belgium: IEEE, 2013, pp. 760–763.

[9] S. van der Meer, “5G & Autonomic Networking - Challenges in closing the Loop,” in *IEEE First International 5G Summit*, May 2015.

[10] S. Achuthan and J. O’Meara, “A system for monitoring mobile networks using performance management events,” in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, May 2013.

[11] L. Fallon, S. van der Meer, and J. Keeney, “Apex: An engine for dynamic adaptive policy execution,” in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, April 2016, pp. 699–702.

[12] Ericsson. (2016, Feb) Ericsson network manager (enm). Ericsson. [Online]. Available: http://www.ericsson.com/ourportfolio/products/network-manager

[13] S. Feghhi *et al.*, “Diagnosing channel issues using gtp protocol messages in lte core networks,” in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, April 2016.

[14] Ericsson. (2016) Ericsson cloud manager (ecm). Ericsson. [Online]. Available: https://www.ericsson.com/ourportfolio/products/cloud-manager

[15] A. Bondkovskii *et al.*, “Qualitative comparison of open-source sdn controllers,” in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, April 2016.

[16] S. van der Meer, J. Keeney, and L. Fallon, “Dynamically Adaptive Policies for Dynamically Adaptive Telecommunications Networks,” in *Network and Service Management (CNSM), 2015 11th International Conference on*, Nov 2015, pp. 182–186.

[17] ETSI, “Network Functions Virtualisation (NFV); Architectural Framework,” ETSI, Tech. Rep. GS NFV 002, October 2013.

[18] 3GPP, “Network Architecture,” TS 23.002, 3GPP, December 2012.