Remediating Anomalous Traffic Behaviour in Future Networked Environments

Angelos K. Marnerides, Matthew Jakeman, and David Hutchison

Infolab21, Computing Department
Lancaster University, Lancaster, UK
{a.marnerides,m.jakeman,dh}@comp.lancs.ac.uk

Dimitrios P. Pezaros

Department of Computer Science
University of Glasgow, Glasgow, UK
dp@dcs.gla.ac.uk

Abstract. The diverse characteristics of network anomalies and the different recovery approaches that can subsequently be employed to remediate their effects, constitute static defence mechanisms tuned at responding to specific abnormalities suboptimal for providing an overall resilience framework. The emerging autonomic network environments in particular, require always-on, adaptive, and generic mechanisms that can integrate with the core networking infrastructure and provide for a range of self-* capabilities, ranging from self-protection to self-tuning. In this paper we present the design and implementation of an adaptive remediation component built on top of an autonomic network node architecture [2]. A set of pluggable modules that employ diverse algorithms and explicit cross-layer interaction have been engineered to mitigate different classes of anomalous traffic behaviour in response to both legitimate and malicious external stimuli. In collaboration with an always-on measurement-based anomaly detection component, our prototype empowers the properties of self-optimisation and self-healing.

Keywords: Autonomic networks, resilience.

1 Introduction

The design of autonomic architectures requires the incorporation of self-* properties that will ensure the optimal network operation in the face of the dynamically-changing behaviour of next generation, converged networked environments. The merging of heterogeneous systems and networks within such environments requires that effective measurement and control mechanisms are composed in a multi-dimensional fashion on all three data, control and management planes.

In this paper, we present the design of resilient systems, which are defined to provide and maintain an acceptable level of service in the face of various challenges to normal operation [8]. We particularly focus on the area of network traffic anomalies since they pose great challenges to infrastructure and their confrontment after they are detected is something non-trivial to achieve. This is mainly due to the different types of anomalies triggered either from a legitimate or malicious intent, and whose overall defence cannot be provided by parsimonious models or mechanisms.

Within the context of autonomic communications, anomalies span dynamically across different systems and networks, and hence the employment of recovery mechanisms in order to remedy them encounters hard design and implementation decisions. It is necessary for such mechanisms to be foundationally supported by infrastructures that facilitate flexible design frameworks. The EU FP6 Autonomic Network Architecture (ANA) Project has successfully deployed an Autonomic *node* (ANA *node*) which –via its available development framework and API– allows the design of such complex mechanisms.

Based upon the resilience requirements we have identified within ANA, we have designed an architecture composed by an anomaly detection unit and an anomaly remediation engine [9]. Synergistically and by employing cross-layer information exchange, both components empower the properties of self-protection, self-learning, self-optimisation and self-healing at the onset of anomalous traffic behaviour. The first two are provided by the detection unit, whereas our remediation engine accommodates the latter two properties, and it is embedded within the resilience architecture built on top of the ANA node. Through our algorithmic design, we provide an explicit interaction between the application and the network layers which can provide performance benefits for each layer with respect to recovery from an ongoing anomaly (e.g. Flash Event- FE, DoS). The design complements our previous work [13][11], and our implementation has enabled the instrumentation of diverse remediation strategies within a flexible API such as the one provided by the ANA architecture.

The remainder of this paper is structured as follows: section 2 briefly describes our resilience architecture that hosts the remediation mechanism, and introduces the ANA node infrastructure. Section 3 shows the internal architecture of our remediation framework as well as the algorithmic design and evaluation. Section 4 presents the prototype implementation, whereas section 5 outlines future work and concludes the paper.

2 Resilience Architecture

The resilience architecture is encapsulated as an integral part of an ANA *node* and is composed by two Functional Blocks (FBs); the Detection Engine (DE) and the Remediation Engine (RE). FBs compose one of the most primitive abstractions within ANA and their operations may reside either on the control or the management planes. They can be deployed locally on a single ANA node or be distributed within a compartment. A compartment within ANA is the operational service context where FBs cooperate to provide a service of any type [14].

In practice, an ANA *node* is a microkernel that provides a message switching broker service to Functional Blocks (FBs). It is composed by three core segments: the Minimal Infrastructure for Maximal Extensibility (MINMEX), the ANA Playground and the ANA hardware abstraction layer. The MINMEX is the means for allowing and providing basic low-level functionalities which are required for bootstrapping and running ANA. At the same time it facilitates the generic sets of methods (API) that are used by its "clients" (e.g.. applications, protocols).

The most complex and advanced networking functionalities within an ANA node reside in the ANA Playground. This segment of the node hosts both commodity and bespoke components (e.g. cryptographic primitives, compression schemes, error recovery codes), and it couples a development and execution environment allowing the implementation of FBs.

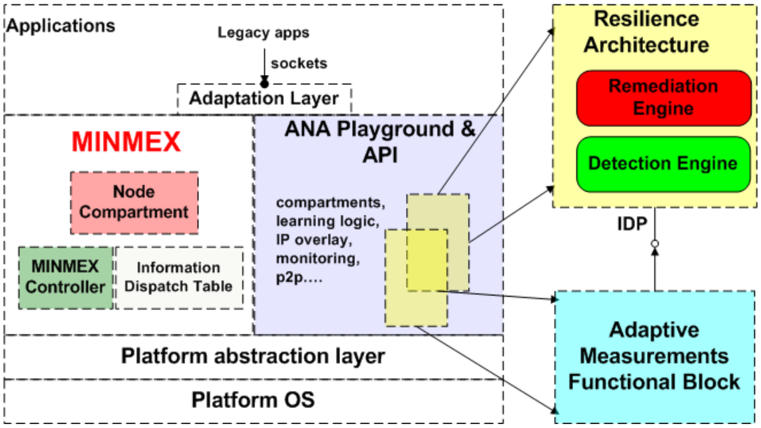

Fig. 1: Resilience Architecture within an ANA node.

Fig. 1 shows our resilience framework which is engineered within the ANA Playground, and follows the API principles provided by it. Since it is a measurement-based framework, the architecture purely depends on the functionality offered by a dedicated Adaptive Measurement/monitoring Functional Block (AM FB) that consists by 5 FBs available via the Playground. Data passing and message interaction between the measurement framework and the resilience architecture is achieved through certain data interfaces which in ANA are referred to as Information Dispatch Points (IDPs).

IDPs act as generic communication pivots between the various FBs and they offer the advantage of re-organizing communication paths between the FBs. IDPs also allow the implementation of forwarding tables which are fully decoupled from addresses and names. For example, in ANA, the next hop "entity" may reside either locally on the system or within a compartment and is always identified by an IDP. Consequently, this allows for the easy addition and usage of new networking protocols and technology, as long as their communication services are exported as IDPs.

There are several IDPs published (e.g. flow reception IDP from the DE; notification reception IDP from the RE; sampling and capturing configuration IDP from the Adaptive Monitoring Functional Block (AM FB)) as services from both the Resilience FBs and the AM's FBs, albeit not included in Fig.1 for the sake of simplicity.

However, the main purpose of this section is to explain the main interaction between the two frameworks and show how resilience can be accommodated.

Initially, the DE receives flow information from the AM FB on a dedicated IDP and internally performs entropy estimation on selected flow features (e.g. packet inter-arrival time, bytecount), in order to provide a prediction about the evolutionary behaviour of the traffic, and to detect possible anomalies. Subsequently, the entropy results are passed to a Supervised Naïve Bayesian classifier which, based on past classified traffic can compare and further categorize a possible anomaly to its correct label (e.g. DDoS, alpha flows, Flash Events (FEs)), and decide whether an anomaly occurs on a local or a compartment-wide scenario. According to the vicinity of the anomaly, the DE is then responsible for notifying the appropriate RE that may reside locally (on the same ANA node) or remotely within the same compartment.

The operations undertaken by the DE FB are composed by two main sub-units which, based on the ANA terminology, are referred to as bricks. Their complex functionalities pose great overall design and implementation challenges which we do not intent to describe in this document, since we are focusing on the explanation of the RE. A detailed description regarding the internals of the DE can be found at [9], [12] and [10].

3 Remediation Engine Design

3.1 Remediation Engine Internal Architecture

The composition of the architecture described below has been a challenging task since we initially had to identify and further evaluate algorithms that would be appropriate as robust remediation strategies. In parallel, we had to consider whether our design was practically feasible to instrument using the ANA API. This subsection mainly presents the engineering aspect of our design followed by the algorithmic evaluation.

Fig. 2 presents the RE internal architecture which is in charge of mitigating the effects of a local or compartment-wide traffic anomaly. The RE is composed by two main functional modules: the Defender and the Messenger. The former executes node-local remediation algorithms, and the latter distributes the instance of an event to remote REs within a network compartment, as required. Apart from the two core bricks, there is also the Configuration Manager (CM) which is an infrastructural unit, and provides for the dynamic binding and configuration of the overall RE with a local or remote DE. Due to the diversity of network anomalies and the different effective remediation strategies that can be deployed according to their nature, we have focused on two broad categories, those of Denial of Service (DoS) –including distributed DoS–, and Flash Events (FEs). We have therefore considered two families of remediation algorithms for our Defender module, namely traffic shaping and dropping (in response to an e.g. DDoS attack), and geographic region-aware clustering and load-balancing (in response to a FE), respectively.

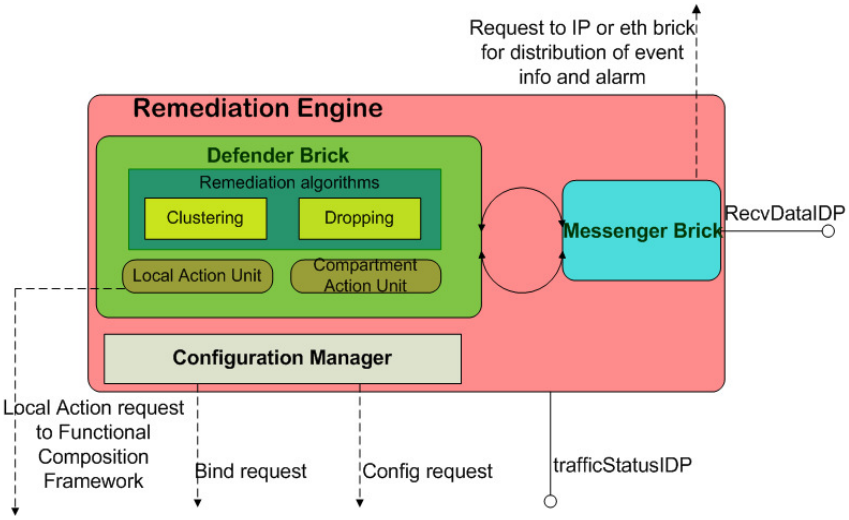

Fig. 2: Remediation Engine Internal Architecture

This latter remediation strategy enforces traffic prioritisation to ensure path diversity and maximises throughput by propagating popular content to geographically diverse areas of the network. It then uses cross-layer information to redirect clients to alternative sources of content, and therefore reduces link stress, effectively providing for network self-optimization [11]. In addition, due to path diversity increasing in-network, this algorithm is inherently distributed, since a RE instructs a peer to take a clustering action on its behalf. To a lesser extent, the same holds for the (D)DoS remediation based on packet dropping, since distribution among multiple REs can be exploited as an efficient pushback mechanism that enhances system and network self-healing, and alleviates the detrimental effects of an attack at the "last mile".

3.2 Remediation Strategies

A core process within our engine's development lifecycle was the algorithm selection and evaluation phase undertaken before the actual prototype implementation. As already mentioned, our intention was to accommodate diverse remediation strategies that would enable self-optimization and self-healing at the onset of two particular types of anomalies: FEs and (D)DoS.

In parallel with region-clustering and load balancing that are considered beneficial for DDoS defense [1], we strongly support that the confrontment of such events is required to include a dropping mechanism. Either on a distributed or local attack scenario, our dropping methodology is collaboratively accommodated with the actions taken by the Functional Composition (FC) framework. The FC as presented in [14] is a core infrastructural unit present in any ANA *node* that leverages self-management capabilities on the data, control and management planes. One of the features of the FC framework is congestion management for which a packet dropping utility is provided. The FC contains the Packet Sink brick which employs the traditional Random Early Detection (RED) algorithm [7], and its services are publically visible within the same network compartment. Therefore, at the onset of (D)DoS attack, our Remediation Engine (RE) sends a notification to the FC containing a list of the malicious source addresses. Subsequently, the Packet Sink brick is notified by the FC and marks the reported packets as "optimal", dropping them immediately in order to block the attack traffic. Since the Packet Sink brick supports RED, we inherit some of its terminology in our scheme stating "optimal" packets as those holding a high dropping preference in the system's queue.

In addition to the dropping mechanism offered by the FC, we have evaluated the gains of diverse remediation strategies for fast content propagation, applicable to event of legitimate, yet adverse requests for content hosted over a single network topology. We have simulated P2P file sharing overlays to demonstrate how application-network cross layer interaction can alleviate the detrimental effects of flooding phenomena such as Flash Events. We have developed two cross-layer services that can be made accessible to an ANA network compartment (as well as to conventional ISP networks), namely a "distance" and a "region-awareness" services. The "distance" service is a facility that simply takes IP addresses as input and returns a distance measurement between the source and the requester. Since, distance may be specified in several ways in ANA, in our case we have used the Autonomous System (AS) path length that returns an AS Proximity (ASP) metric, and then selects the least-AS-distant content provider. "Region-awareness" is a service that also takes a set of addresses as input, and returns these addresses clustered to several subsets according to the different paths traversed between two nodes [13].

We have developed a number of augmented and region-aware overlay algorithm variants in order to demonstrate how operation can be optimised using explicit cross-layer interaction. The augmented overlay algorithm uses the "region-awareness" service so that providing peers can load-balance their response traffic. When a provider reaches its simultaneous serving threshold, it explicitly redirects further clients to the subset of peers (providers) it has already served through the same "regional" cluster that the incoming request came from. Using this strategy, providers tend to spread the overlay traffic load to diverse segments of the underlay infrastructure. In addition, we have developed variants of a Region-aware Overlay (RegO) algorithm which, in contrast to the augmented algorithm, uses a central tracker facility to provide requesters with all currently serving overlay peers [13]. The RegO implements Random (Ran) provider selection and region –aware load balancing to requests.

Within our simulations both algorithms have employed two variants of clustering namely, Simple Clustering (SC) and Hierarchical Clustering (HC). In SC, the server clusters requests to segments that are based on the first-hop egress link traversed by the response traffic. Simply enough the total simultaneous request threshold defined as T is divided by the servers' v egress links and $\tau_i = T / v$ simultaneous requests are served per-cluster. The HC variant accommodates a hierarchical clustering of the requests based on the traversed response traffic from both the first and second hops. Under the HC scenario the initial threshold T is divided by the number of first-hop egress links v to produce *i* first-hop thresholds $\tau_i = T / v$, each of which is further divided by the number of egress links attached to first hop *i*.

We have used NS2 [15] and BRITE [4] to construct numerous power-law AS-level topologies to include 100, 500, and 1000 AS nodes, each having a minimum degree of 2, 3 and 4 links per AS leaf [3][5]. The gains of the explicit underlay/overlay performance were assessed under the scenario of spreading the so-called first chunk (in our case 1MB) of content among participating peers which serve at most 10 simultaneous transfers each. The performance metrics used were the individual transfer throughput in KB/s and the maximum link stress over the complete Internet-wide topology. Fig. 3 shows the effect of cross-layer algorithms in increasing transfer throughput over different AS-size topologies and their respective overlays.

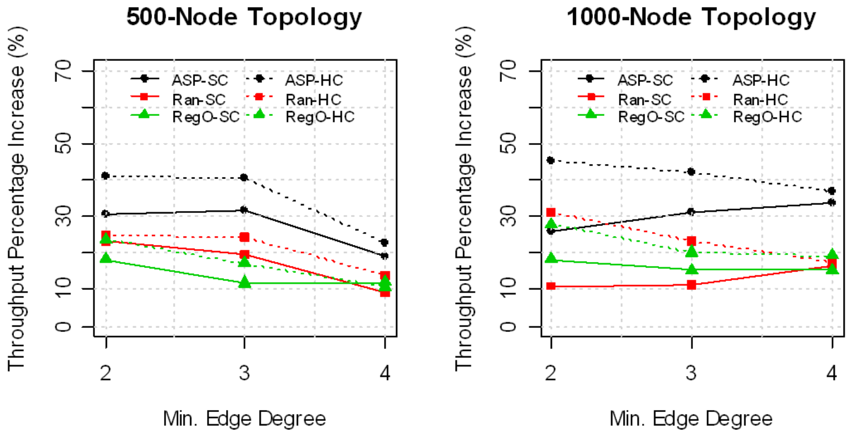

Fig. 3: Percentage increase in mean individual transfer throughput for cross-layer algorithms.

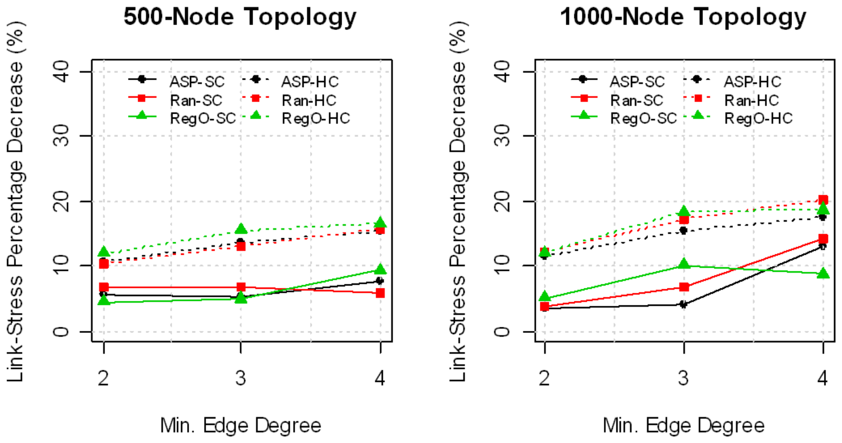

Fig. 4: Percentage decrease in topology-wide maximum link stress for the cross-layer algorithms.

Variations in throughput increase with respect to the minimum access edge degree of the topologies are also shown. The solid lines represent the performance gains of the cross layer-algorithms with first-hop SC, and the dashed lines show their second hop HC counterparts.

The decrease of maximum link stress for the same AS-level topologies as for those in Fig. 3 is presented in Fig. 4. Even though there is no clear correlation between throughput and link stress, it is evident that on average cross-layer algorithms outperform their simple overlay counterpart. It is evident that every algorithm employing hierarchical clustering on the requesting peers consistently outperforms simple clustering based on network access link. In addition, the augmented algorithm with ASP provider selection presents significant gains over the rest of the algorithms with a 50% optimisation. A quite appealing general observation is that the length of the content providers list (i.e. the number of alternative sources) is not of major importance neither for reduced link stress nor for increased throughput transfer.

4 Remediation Engine Implementation

The Remediation Engine (RE) is composed of two main entities, the Defender and the Messenger. These are both implemented in the form of ANA bricks, the most atomic elements of functionality within a FB. Fig. 5 displays a conceptual overview of the data communication that takes place once an anomaly has been detected by the DE.

The Defender publishes the *rmInfoIDP* which is responsible for receiving anomaly information by the Detection Engine (DE).

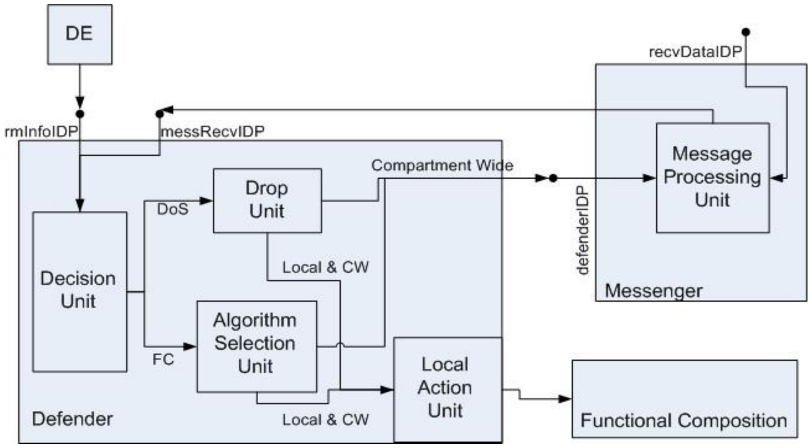

Fig. 5: RE data flow & functionality.

The information the DE sends a structure containing the result of the Bayesian classification along with an indication as to whether the abnormality has a local or compartment-wide impact. Subsequently, this information is passed on to the Decision Unit (DU) that resides within the Defender.

The DU is in charge of processing the information received from the DE and decides whether the anomaly has been classified as a (D)DoS attack or a FE. Subsequently, it forwards the information to the appropriate processing unit.

Acting as an integral part of the Defender, the Local Action Unit (LAU) is responsible for informing the Functional Composition (FC) framework what actions to take based on the nature of the attack. This is achieved by constructing a LUA [6] message that the FC can use to insert appropriate traffic filters. If the anomaly is a DoS attack, the DE information is passed to the Drop Unit whose responsibility is to inform the Local Action Unit (LAU) that packets need to be dropped. In cases where the attack has been classified as local, the LAU still needs to interact with the local FC instance in order to inform it that local dropping is required. In addition, the LAU provides a list of the malicious source addresses to the FC which via its Packet Sink brick marks them as "optimal" in order for the Random Early Detection (RED) mechanism employed to drop them immediately. In parallel with dropping, the FC constructs a filter based on the listed source addresses and triggers its congestion control utilities with an initial packet inspection. Similarly in the case of a compartment-wide attack (i.e. DDoS), it is necessary for the LAU to inform the Messenger using its *defenderIDP* so the Messenger can distribute information about the anomaly to all REs within the same network compartment.

In the scenario of a FE, the anomaly-related information is passed on to the Algorithm Selection Unit (ASU). The ASU is the unit in charge for performing decisions about which of the algorithms discussed previously in this paper to be employed by the Remediation Engine (RE). Once a decision is made, and since the FE is a compartment-wide phenomenon, the ASU informs the LAU in order to construct an appropriate LUA message and send it over to the local Messenger instance.

The primary purpose of the Messenger is to disseminate information about an anomaly to other nodes in a compartment if the anomaly was classified as not just affecting the local host. When it receives a message over the *defenderID*, it checks to see what type of anomaly has been detected to enable it to construct an appropriate message to send to other Messengers in the same network compartment. Messengers also transmit Autonomous System Proximity (ASP) metrics as well as routing information (i.e. next hop) between themselves since this information is required to be used by the region-aware and load balancing algorithms as deployed by the ASU. When next hop load balancing is being performed, the Defender can inform the Functional Composition Functional Block (FC FB) where to forward traffic based on the next hop and the FC FB can place appropriate filters to route the traffic.

In a DDoS attack event, the Messenger also constructs a new message containing a list of source addresses that have been identified as originating nodes in the attack. A parallel process performed by the Messenger is to resolve all Messenger-compatible reception data IDPs (i.e. *recvDataIDP)* that are public and visible within the same compartment so it can broadcast this message to other Messenger bricks. When another brick of the same type receives this message, it is then eligible to inform its local Defender that it is required to remediate the attack. This action is achieved from an internally viewed IDP namely the *messRecvIDP*. As soon as this information is passed to the Defender, the appropriate local action is triggered (i.e. dropping via the FC FB) in the same way it does when informed of a local DoS by the DE.

Similarly, at the onset of a FE, the Messenger interacts with other REs via their Messenger bricks and sends a message in the compartment informing it about the FE along with a list containing the addresses that are directly related to the phenomenon. The Messenger additionally transmits a notification stating the algorithm that the ASU decided as appropriate for compartment-wide deployment. For instance, under the circumstance where a FE has an impact on next hop nodes within the same compartment, the ASU decides and informs the Messenger to notify the next hop Messenger instances that the most suitable clustering algorithm for all of them to collaboratively perform for confronting the phenomenon is the Region-aware Overlay - Simple Clustering variant (RegO SC). Subsequently any Messenger that receives this information then pass the algorithm selection request on to its local Defender which triggers the requested scheme

5 Conclusions & Future Work

In this paper, we have presented the design and implementation of a traffic anomaly remediation component that can be an integral part of next generation autonomic network infrastructures.

Through our design and implementation, we have demonstrated that the correct exploitation of carefully designed infrastructures enables complex issues such as the remedy of network anomalies to be effectively resolved. Our remediation framework prototype contributes towards the instrumentation of diverse remediation methodologies and empowers core autonomic properties such as self-optimization and self-healing.

After the promising results obtained through large-scale simulation, ongoing work focuses on testing our framework under actual operational conditions and traffic scenarios. Our intention is to evaluate our remediation mechanisms within the overall resilience architecture that we have also presented in this paper. The evaluation will be conducted in the autonomic communication testbed (ANA-Lab) provided within the ANA project. The ANA-Lab offers the capability of virtual topology instrumentation of ANA nodes through distributed monitoring and control facilities. Our main objective is to examine the practical system performance of our prototype through experiments using live as well as pre-captured operational traffic traces

Acknowledgments. The authors would like to thank the EU/FP6 IST Autonomic Network Architecture (ANA) for its partial support within this piece of work.

References

[1] Asosheh A., Ramezani N., “A Comprehensive Taxonomy of DDoS Attacks and Defence Mechanism Applying in a Smart Classification.”, WSEAS Transactions on Computers, Vol. 7, Issue 7, p 281-290, April 2008

[2] Autonomic Network Architecture (ANA) Project details available at: http://www.ana-project.org

[3] Barabasi A., L., Albert, R., “Emergence of scaling in random networks.”, Science pages 509-512, October 1999

[4] Boston University Representative Internet Topology Generator (BRITE), available at: http://www.cs.bu.edu/brite

[5] Bu, T., Towsley, D., “On distinguishing between Internet power law topology generators.”, IEEE INFOCOM’ 02 , New York, USA , June 23-27, 2002

[6] De Figueiredo, L., H., Ierusalimschy R., Celes, W., “LUA: An Extensible Embedded Language.” Journal of Software Tools, Vol 21, Num. 12, 1996, National Center for Biotechnology , Information, http://www.ncbi.nlm.nih.gov

[7] Floyd S., Jacobson V., “Random Early Detection gateways for Congestion Avoidance.”, IEEE/ACM Transactions in Networking, Vol. 1 p 397-413 , August 1993

[8] Hutchison, D., Sterbenz, J. P.G, Jabbar, A. Sholler, M., 2006 D3.2: Resilience/Security Framework, Deliverable D3.2 ANA December 2006

[9] Marnerides A. K., Pezaros D. P., Hutchison D., “Detection and Mitigation of Abnormal Traffic Behaviour in Autonomic Networked Environments”., 4th ACM SIGCOMM CoNEXT Student Workshop, December 9-12, 2008, Madrid, Spain.

[10] Marnerides A., K., Pezaros D.,P., Hutchison D., “Autonomic Diagnosis of Anomalous Network Traffic”, 4th IEEE WoWMoM Workshop on Autonomic and Opportunistic Communications (AOC 2010), Montreal, Canada, 14-17 June 2010

[11] Pezaros, D., P., “Cross-Layer Optimisation of Network Response at the Onset of Bursty Requests.”, in Proceedings of Multi-Service Networks (MSN'06), Cosener's House, Abingdon, UK, July 13-14, 2006

[12] Pezaros, D., P., Marnerides A., K., Hutchison D., 2008 D3.10: Measurement-based Resilience Mechanisms, Deliverable D3.10 ANA December 2008

[13] Pezaros, D., P., Mathy, L., Explicit Application-Network Cross-layer Optimisation, 4th International Telecommunication NEtworking WorkShop (IT-NEWS) on QoS in Multiservice IP Networks (QoS-IP 2008), Venice, Italy, February 13-15, 2008.

[14] Sifalakis M., Louca A., Peluso L., Mauthe A., Zseby T., "A Functional Composition Framework for Autonomic Network Architectures ". In proceedings of 2nd IEEE International Workshop on Autonomic Communications and Network Management (IEEE NOMS/ACNM '08), Salvador, Bahia, Brazil, April 7-11, 2008.

[15] The Network Simulator 2 (NS2) available at: http://www.isi.edu/nsnam/ns/