TRONE: Trustworthy and Resilient Operations in a Network Environment

António Casimiro*, Paulo Veríssimo*, Diego Kreutz*,
Filipe Araujo†, Raul Barbosa†, Samuel Neves†, Bruno Sousa†, Marília Curado†,
Carlos Silva‡, Rajeev Gandhi§, Priya Narasimhan§
*University of Lisbon, Portugal
{casim, pjv}@di.fc.ul.pt, kreutz@lasige.di.fc.ul.pt
† University of Coimbra, Portugal
{filipius, rbarbosa, sneves, bmsousa, marilia}@dei.uc.pt
‡Portugal Telecom
c-miguel-silva@telecom.pt
§ Electrical & Computer Engineering Department, Carnegie Mellon University, USA
{rgandhi, priya}@ece.cmu.edu

***Abstract*—Cloud infrastructures play an increasingly important role for telecom operators, because they enable internal consolidation of resources with the corresponding savings in hardware and management costs. However, this same consolidation exposes core services of the infrastructure to very disruptive attacks. This is indeed the case with monitoring, which needs to be dependable and secure to ensure proper operation of large datacenters and cloud infrastructures. We argue that currently existing centralized monitoring approaches (e.g., relying on a single solution provider, using single point of failure components) represent a huge risk, because a single vulnerability may compromise the entire monitoring infrastructure.**

In this paper, we describe the TRONE approach to trustworthy monitoring, which relies on multiple components to achieve increased levels of reliance on the monitoring data and hence increased trustworthiness. In particular, we focus on the TRONE framework for event dissemination, on security-oriented diagnosis based on collected events and on fast network adaptation in critical situations based on multi-homing application support.

To validate our work, we will deploy and demonstrate our solutions in a live environment provided by Portugal Telecom.

***Keywords*-Cloud Computing, Trustworthy Monitoring, Intrusion Tolerance, Root-Cause Analysis**

I. Introduction

The Telecom industry is going through rapid changes leading to what are commonly designated as Next Generation Networks (NGN): different technologies converging into a network tissue able to provide multiple services with on-demand provisioning, in a seamless and technology-independent manner.

The introduction of new technologies and new services, especially at higher levels of abstraction, pushes the demand on infrastructure to new levels, increasing the likelihood of failures, either accidental or malicious. On the other hand, users are each day more demanding in terms of the quality of the service they get. The trend is deviating from sheer performance ("fast") to meeting expectations about how well the service is provided against what was promised ("trustworthy"). In this sense, the competitiveness of a telecom operator is drastically influenced by the way it masters a few key factors: fast service innovation, high quality of service (QoS), quality of protection (QoP), and, finally, high operational efficiency and agility.

In order to achieve these objectives in the near and mid-term future, we emphasize the need for reducing hazards, both proactively, by increasing architecture robustness, and reactively, by improving the means for detection and recovery from anomalous situations like faults and attacks. In this position paper we focus on a set of well-defined problems related to trustworthy network operation, and describe the respective solutions that are being developed in the scope of the TRONE project. More specifically, we focus on the following three problems:

- **Architectural robustness.** Network and infrastructure monitoring is essential for management and administration tasks. Therefore, the monitoring infrastructure itself must be made robust to both accidental and malicious faults. We propose a Fault and Intrusion Tolerant (FIT) event broker, which can be used to manage all the event flows carrying monitoring data from sensors to monitoring consoles. This FIT broker is robust against malicious faults through the use of Byzantine fault tolerant broker replicas, and may be used to disseminate monitoring data to several systems using a publisher/subscriber approach, thus supporting replicated monitoring activities.
- **Automated failure diagnosis.** As the scale and complexity of the monitored infrastructures grows, it becomes increasingly difficult to perform accurate diagnosis. In many cases, human intervention and inspection of logs or monitoring data becomes necessary to determine what the actual problem may be. We propose

new approaches for failure detection and prediction, and for problem determination (automated fingerpointing). These will allow for improved automation and proactive behavior, rather than slow reactive behavior in response to failures.

- **Fast reconfiguration.** When offering Infrastructure as a Service cloud services, the network should perform well despite overloads resulting from failures (or attacks affecting some links). Fast reconfiguration when some problems are detected is hence an important asset. We propose a multi-homing approach to achieve improved resilience to faults affecting the communication links used by client applications, which can make use of security-related diagnosis information to trigger reconfigurations.

While the described solutions are generic enough to be useful in diverse contexts, we introduce the concrete example scenario that sets the scope for the work being developed. The scenario is provided by the industrial partner in the project, Portugal Telecom (PT), and concerns the infrastructure for PT's cloud. The increasing demand for cloud resources (from PT's clients and internally within PT) stresses the requirements on the needed trustworthiness of the operation, and specific concerns with scenarios of attacks and intrusions which might disrupt the infrastructure monitoring, operation and orchestration. The scenario motivates the need for the proposed solutions and it will also provide the necessary use cases to validate the developed solutions.

Given the above, the paper is structured as follows. We start, in the following section, by introducing the considered cloud computing infrastructure scenario provided by PT. Then we provide an overview of the overall TRONE framework, explaining how the different aspects of our work are integrated in the considered scenario and contribute to the increasing the trustworthiness of the overall operation. In sections IV, V and VI we describe our contributions for architectural robustness, automated fault diagnosis and fast reconfiguration using multi-homing. Finally, we conclude the paper, referring to the next steps in this work.

II. Portugal Telecom's Cloud

Portugal Telecom (PT) has been making a great effort to implement cloud-based solutions. To improve performance, security and availability, PT evolved from a small deployment with virtualized operating systems, to virtualize a large number of server machines. This new environment is based on a complete virtualization infrastructure, including the datacenter network core, the computational power, and the storage area network.

Currently, PT is using its cloud environment to offer several major services, both to enterprise customers, and for internal needs as well. These services significantly raised the level of cloud expertise of the PT's IT personnel, thus generating economies of scale in several other services, like storage and backup services, network and security and, of course, public and private servers. It is clear for us that this trend is still growing. In the next few years, cloud computing environments will most certainly be the most important topic in the Portugal Telecom services portfolio. Even other innovative and important services, like IPTV, are likely to run over the cloud infrastructure.

A. Environment Description and Threats to the Cloud

Portugal Telecom's current cloud computing infrastructure is implemented on top of two different virtualization technologies: VMware ESX and Parallels Virtuozzo. Both current infrastructures are becoming a scarce resource, due to the amount of requests (both internal and external) being processed. To overcome this limitation, PT is deploying a completely new infrastructure for cloud computing services, based on VMware (virtualization), Cisco (network and security) and EMC (storage) technologies.

Given the growing dependence of PT on cloud-based technologies, security of this kind of infrastructures is paramount to the company. In fact, two great concerns appear in any virtualization infrastructure: i) the possibility of an attacker to compromise the hypervisor and take control of the virtualization infrastructure, and ii) the possibility (already real in several other infrastructures of this kind in different parts of the world) of a malicious utilization of the cloud computing infrastructure to execute a series of network attacks against the infrastructure or the others using the computational power layer. In the TRONE project, we are particularly interested in evaluating the threats against the hypervisor. These are of crucial importance for the cloud infrastructure, as malicious users can take control over other guest machines, if they manage to exploit some fragility in the hypervisor. For example, they may be able to read memory outside of their own space, or to escalate privileges inside their physical machine. Other problems are of interest to us: attackers (or even non-malicious users) may cause harm without controlling hypervisors, as they may load their own virtual machines to the point where they cause service disruption or degradation in other guest virtual machines.

The network and storage platforms are also under careful consideration, as there can be attacks against the maximum number of Input/Output Operations Per Second (IOPS) allowed in the storage, or attacks like routing cache poisoning. The implemented controls can also be considered as an attack vector if segregation controls are bypassed, or access to privileged data is accomplished, or even if attackers manage to use a very large share of the bandwidth.

B. Event-Based Monitoring of the Cloud

To tackle these and other possible vulnerabilities in the PT's cloud environment, we need to rely on a series of control mechanisms. First, we monitor traditional attacks that can detect resource exhaustion, like CPU, Memory,

Disk Space, etc. Additionally, we are working on hypervisor monitoring and verification techniques, based on heart-beats or some similar mechanism, to guarantee the correct and timely execution of the hypervisor. Another important control mechanism that we intend to use is the implementation of IDS/IPS techniques, capable of detecting network attacks. These are based on signatures and heuristics-based detection engines, which can detect attacks or attempts to evade detection techniques. Clearly there is the need to execute a characterization of the network usage profile for each guest virtual machine, with strong correlation procedures that can enable the discovery of covert channels, stealth attacks or control procedures that can command other systems to execute attacks.

To enable proper operation of all these mechanisms, the TRONE project proposes an event-based monitoring platform, capable of correctly receiving and processing all the events that it gets from the physical and virtualized infrastructure, even under the presence of byzantine failures.

III. The TRONE Approach

We aim at addressing some of the problems described in the previous section, to improving the overall trustworthiness of the operation and management of large datacenters and cloud computing infrastructures, such as the one introduced in the previous section. Figure 1 provides a general view of our focus areas in the context of the considered scenario, which we address in the remainder of the paper.

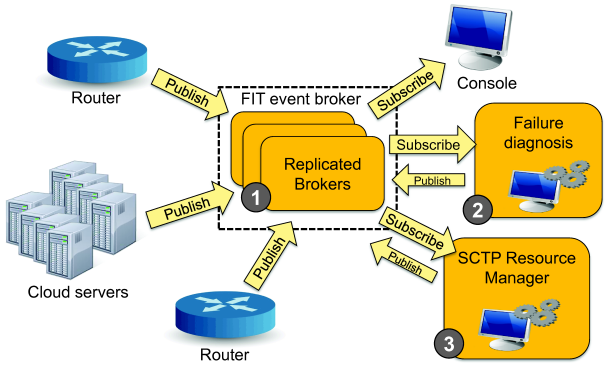

Figure 1. TRONE blocks for trustworthy monitoring.

Monitoring information is produced in the form of events that are delivered from the monitored to monitoring components and systems. The cloud computing infrastructure involves a wide range of equipments, which are represent in the figure by the cloud servers. Routers represent networking equipment, which are also monitored. Many event flows need to be set up to all the relevant monitoring systems, which may range from simple consoles to complex correlation engines. In order to deal with all these event flows, we propose a Fault and Intrusion Tolerant (FIT) event broker (1 in the figure), following a publisher-subscriber paradigm for event dissemination. It is fundamental to ensure that this event broker is resilient to crash faults and, in certain cases, to Byzantine faults. For that purpose, we use replicated brokers and follow the State Machine Replication in the design of our solution.

Using the FIT broker, events are published and delivered to all interested subscribers. Event subscriptions are necessary, in particular, for performing automated failure diagnosis and prediction of problems within the cloud infrastructure (2 in the figure). Detecting failures in large-scale distributed systems is challenging, as modern datacenters run a variety of applications and systems. Current techniques for detecting failures often require training to detect failures, have limited scalability, or are not intuitive to sysadmins. In TRONE we address the failure diagnosis problem by proposing a lightweight and scalable algorithm for distributed systems which detects failures using only correlations of operating system metrics collected transparently. We aim to be effective in detecting failures in our target systems, minimizing false-positives while detecting as many failures as possible. This will be an important contribution to increase the quality of protection of Portugal Telecom's cloud infrastructure.

Finally, we also focus on the problem of fast and automatic reaction to network failures and degradation. The approach relies on the use of the Stream Control Transport Protocol (SCTP), which provides support for resilience, namely through multi-homing. Detection of problems is based on the ability to monitor communication channels and collect communication-related information from SCTP-aware client applications (3 in the figure). As shown in the figure, some of the monitoring services may also publish events, for instance containing diagnosis information. This allows for further correlation activities to be performed by other components subscribing to the event channels in which these events are published.

IV. FIT event broker

The objective of the FIT event broker is to provide a replication-based reliable and trustworthy communication layer for transporting event messages from probes to consoles. Replicas execute as deterministic state machines with respect to operations that affect their state (e.g., channel creation and new subscriptions), and are able to forward published (event) messages to respective subscribers, while providing varied levels of resilience and quality of service for each event channel. Publishers (probes, agents, sensors) are responsible for creating messages regarding event occurrences and publishing of event channels supported by the FIT event broker. Subscribers (consoles, event correlation engines, resource managers) are responsible for receiving event messages in order to perform their tasks. We adopted the publisher/subscriber model for communication and event handling [1], providing topic-based message routing [2].

The solution was designed to address the Quality of Service (QoS) [3] and Quality of Protection (QoP) [4] requirements of the considered environments. QoS refers not only to sheer performance, but also to other qualities like the ordering of events (which might need to be consistent across several subscribers), ability to deal with urgency (delivering first events from some event channels) and persistence of events. QoP refers to the different levels of resilience that might be chosen for different types of events. Both QoS and QoP are attributes of event channels, which are fundamental abstractions in the FIT broker.

The considered fault model directly determines the achievable QoP. For instance, if one assumes that broker replicas can only fail by crashing, then protocols to implement crash resilient event channels will be used, which will only be resilient to the assumed (crash) faults. Our FIT broker provides selectable resilience to crash or to Byzantine faults affecting its replicas. Given the different complexity of the underlying protocols and solutions to implement the event channels, and implied trade-offs with performance, it might be desirable to use crash fault-tolerant (CFT) channels for non-critical events, and Byzantine fault-tolerant (BFT) channels for critical ones. Events sent through CFT channels are essentially forwarded (through replicated paths) and immediately delivered as soon as received by a subscriber, which allows for high throughput. On the other hand, events sent through BFT channels need to be delivered correctly, even if one (or some) replicas are malicious, which requires some voting. Additionally, we use BFT state-machine replication in stricter cases, such as when registering channel subscribers to multiple event broker replicas or when the events in some event channel need to be sent strictly in the same order to all subscribers (e.g. replicas of a correlation engine).

Further to QoP, integrity and confidentiality techniques can be applied through the selection of one of several methods. Simple identification (SI), hash functions (HA), public key signatures (PKS) and public key signature with encryption (PKE) can be provided.

The event message flow, from publishers to subscribers, is presented in Figure 2. When an event channel is created, this is done in all service replicas in a consistent way, and the associated configuration properties are managed based on a control table. This control table contains, for each channel, a list of authorized clients (publishers and subscribers), a topic (channel TAG) and the channel CLASS (which defines the channel QoS and QoP).

One overall design goal is to allow for an easy integration of the FIT broker in the cloud monitoring environment. We provide the means for monitoring tools event messages to be encapsulated in FIT broker messages, which are then transmitted in our event channels. Such datagram encapsulation is done transparently at the provided interfaces. For example, the Nagios monitoring system is composed by probes and consoles that can use our event broker to transparently forward event messages from probes to consoles. The FIT event broker provides simple and usable interfaces to turn it into a practical solution. Clients (publishers and subscribers) have access to a simple and well defined API, providing functions to register, subscribe, publish and receive events from the event broker.

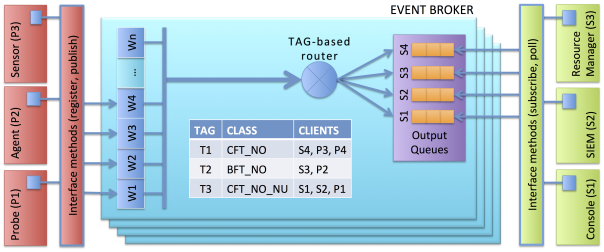

Figure 2. Broker internal components and event message flow.

V. Security-Oriented Diagnosis

Debugging performance problems of cloud-computing programs, e.g., in Hadoop, is difficult due to their scale and distributed nature. The increasing scale of cloud clusters necessitates automating diagnosis of performance problems, as inefficient large programs waste compute resources, and the large scale of these clusters renders manual debugging infeasible. Currently, cloud-computing applications can be debugged by examining its per-node logs of execution. However, these logs can be large, and must be manually synthesized across nodes to debug system-wide problems. Alternatively, such programs can be debugged using profiling/optimization tools such as `jstack` and `jprof`, that are specific to the Java programming language.

In the context of TRONE, we aim to: (i) expose cloud behavior that results from the application's automatic execution, that affect program performance and security, but is neither visible from nor exposed to user-level/application-level code, and (ii) to expose aggregate and dynamic behavior that can provide different insights, (iii) to enable causal tracing of all program data and control flows in a cloud application, i.e., to enable users to trace execution from outputs back up each to the source; this enables detailed what-if analysis to be carried out on cloud applications to perform bottleneck/security analysis and trace deviations to their source, and (iv) to be transparent to the cloud infrastructure and its applications, without requiring additional instrumentation of the cloud platform, by either the system administrators or the application programmers, thereby ensuring that our techniques can work in production environments.

We assume that the OS-level metrics (such as those that the FIT broker gathers transparently from the system/applications) faithfully capture the system's execution state, i.e., all relevant events of interest as defined in our model are logged, with no missing nor additional (spurious)

events. We assume that the timestamps on each host's locally-generated data accurately reflect the order in which the statements are logged. However, we do not assume globally synchronized clocks as we use unique identifiers to causally correlate events.

We propose to determine if a security attack has happened and to identify the culprit nodes, in a cloud application, based on the key insight of peer-similarity among nodes in cloud applications: (1) the nodes (that we loosely regard as "peers") in a cloud cluster/application tend to behave symmetrically in the absence of performance problems, and (2) a node that behaves differently from its peer nodes, is likely to be the culprit of a security attack. We have previously successfully used this approach in diagnosing performance problems in MapReduce/Hadoop-based cloud clusters, and have used it successfully in production environments.

Training: We can develop training profiles of the system based on the injection of security incidents and studying the resulting data that we can gather from the system via the FIT broker. We can use a period of a priori training to summarize system metrics into a small number of behavior profiles. This enables it to extract meaning out of black-box data typically devoid of semantic information.

Localization/Diagnosis: First, we classify each node's OS-metric vector into one of the learned K distinct profiles. We then compute the histograms of these profiles for each node, decayed exponentially with time to weight past behavior less. We then compute the pairwise distance between the histograms of every pair of nodes in the cluster in each time period, using a statistical measure (such as the the square-root of the Jensen-Shannon divergence). If a specific node's histogram differs significantly (by more than a threshold value) from those of a majority of the other nodes, then we declare the system to be undergoing a problem and indict that node to be the culprit.

Root-Cause Analysis: In order to diagnose the root cause of a security incident, we might need access to multiple alternative perspectives of the system, and to fuse these perspectives in a meaningful way to provide an overall "picture" of the system's behavior. We propose to incorporate white-box (application-level) data, if available, in order to determine the root-cause of the problem. We also propose to incorporate network-level trace data to provide a network-level perspective that can be fused along with OS-level perspectives to determine the unfolding of a problem in the distributed system.

A. Visualization

To better understand the obstacles to diagnosis, we interviewed cloud-computing users and system administrators of a large academic cluster (4000 processors, 30PB of storage). After conducting several interviews with system administrators, we discovered that they were already efficient at diagnosing infrastructural performance problems, e.g., disk failures or network congestion, due to their years of experience. The administrators were less familiar with application-level problems and the unfolding of security incidents with respect to cloud computing platforms. We proposed to develop an improved diagnostic interface that should be to support cloud users so that they do not have to approach the system administrators every time they suspect a problem.

At present, the users diagnose problems by monitoring resource consumption, and browsing application-level logs using disparate web-based interfaces. We observed that users had to constantly switch between interfaces to perceive connections between the data and draw conclusions about the fault state of the system. Moreover, users are often presented with stacks of graphs that co-mingle meaningful information with irrelevant data.

Motivated by these problems and our user studies, we have developed visualization tools that operate on volumes of data and extract/display useful patterns of data in an intuitive way to administrators and programmers. Our cloud-diagnostic visualizations have included large-scale heat maps, swim-lane visualizations that show/compare task-level progress across nodes, and finally, the causal dependencies across nodes for both data blocks and tasks. We have already validated some of our visualizations for performance problems, and are extending them to cover security incidents.

VI. Multi-homing for Fast Reconfiguration

Modern wireless devices can integrate different cellular radio access technologies into multi-band cell phones to realize global reachability and ease migration. High-end servers incorporate multiple physical interfaces that can connect to specific networks, or that can be activated in the occurrence of certain events, such as congestion or link failure. In this aspect, multiaccess network selection can be triggered by network events, but requiring the input from the user, as static policies must be configured to allow the activation of interfaces [5]. Resilience support in these scenarios is not scalable and not automated. For instance, nodes cannot react to certain events, unless they have been configured to do so. TRONE enables an enhanced resilience support by employing multi-homing mechanisms that allow fast and intelligent reconfiguration under failures or security events. For instance, if a failure is detected on a network interface, TRONE reconfigures services to other working and secure interface. Resilience can be enabled at different layers, namely, at the physical layer with redundant connections, at the network layer with logical connections, at the transport layer with multiple associations or at the application layer. This allows optimization of protocols for different protection models: the primary-backup model, or the enhanced concurrent model. The former includes mechanisms to detect and monitor all existent interfaces/routes

to a destination and selects one as the primary path, while the remaining are employed as backup. The latter, besides supporting the primary-backup characteristics, enables to use simultaneously multiple interfaces/routes to the destination, even if interfaces/routes have heterogeneous features.

Stream Control Transport Protocol (SCTP) [6] is one of the first transport protocols to incorporate resilience mechanisms. SCTP, natively, supports the primary-backup model, where the primary path is used to forward data. Backup paths are monitored using heartbeat signalling messages. SCTP includes failure detection mechanisms to detect endpoint failures (e.g., via message retransmission) and also to detect path failures (e.g., via retransmissions and timeouts). The multi-homing support of SCTP can be enhanced by using the Application Programming Interface (API) to configure its internal behaviour. Configurations, as the number of retransmissions, or the address to use as primary are just an example of the possibilities that can be explored to perform fast reconfiguration. SCTP, due to its multi-homing characteristics is also the base transport protocol for the Reliable Server Pool [5], which allows servers of a specific service to participate in a pool. Service redundancy is transparent to users. TRONE includes SCTP in its architecture for the multi-homing support, and for the configuration flexibility. For instance, a TRONE intelligent agent can configure the delivery mode of SCTP (e.g., connectionless like UDP or connection-oriented like TCP) by using the SCTP API. TRONE also uses the SCTP API to collect information regarding SCTP, for instance new associations, address changes and values of negotiated timers. This type of information is combined with information of network events (e.g., new interfaces available) and trust information by the TRONE intelligent agent to score each available path. This score is performed automatically, without user input, as a monitoring function is included in the agent. For instance, network parameters such as one way delay, jitter and loss are monitored, as well as all the events associated to network interfaces. With the score function, the TRONE intelligent agent instructs SCTP to use certain addresses. For instance, address A can get a higher score in comparison to address B. As the score formulation includes multiple criteria, it is more robust and efficient in comparison to the reachability metric of SCTP. The TRONE agent, by including trust information, can prevent malicious users, previously detected by security algorithms, to use cloud services. In these cases, TRONE explicitly configures SCTP to prevent association with such users.

VII. Conclusions

This paper describes the main ideas and the approach taken in the TRONE project to achieve trustworthy monitoring. We are presently developing a FIT broker to cope with malicious faults, possibly orchestrated by attackers. The FIT broker uses multiple redundant paths and implements Byzantine fault tolerant consensus to ensure connectivity and correctness of critical events. It also supports other QoS requirements. As a result, the FIT broker is the core component which serves diverse monitoring and management components, such as the multihoming-based reconfiguration management and the automated diagnosis components.

Multihoming gathers information on the state of remote nodes, which is centrally used to decide the most adequate routing alternatives. Our approach enhances resilience by applying multihoming mechanisms that allow fast and reliable reconfiguration in the presence of failures or security alerts.

Given the amount of information that is available in a cloud infrastructure, there is the need to collect, consolidate, and analyze all the events and diagnose the root-cause of any problems. This is an issue of fault location/diagnosis as well as one of visualization. To cope with these problems, the root cause of each incident is diagnosed using events received from sensors through the FIT broker. The consolidated information will be available for visualization by the administrators, aiming at maximizing their ability to handle security incidents.

We intend to apply these solutions together in the cloud computing environment provided by Portugal Telecom in order to demonstrate that improved trustworthiness can be achieved without sacrificing important properties such as practicality and scalability.

Acknowledgment

This work has been supported by the project CMU-PT/RNQ/0015/2009, TRONE — Trustworthy and Resilient Operations in a Network Environment.

References

[1] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Survey*, vol. 35, pp. 114–131, June 2003.

[2] T. Milo, T. Zur, and E. Verbin, "Boosting topic-based publish-subscribe systems with dynamic clustering," in *Proceedings of the ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM, 2007, pp. 749–760.

[3] N. Seth, S. G. Deshmukh, and P. Vrat, "Service quality models: a review," *International Journal of Quality and Reliability Management*, vol. 22, pp. 913–949, 2005.

[4] Foley, Simon N.; et. al., "Multilevel security and quality of protection," in *Quality of Protection*, ser. Advances in Information Security, D. Gollmann, F. Massacci, and A. Yautsiukhin, Eds. Springer US, 2006, vol. 23, pp. 93–105.

[5] B. Sousa, K. Pentikousis, and M. Curado, "Multihoming Management for Future Networks," pp. 505–517, 2011.

[6] J. Eklund, K. Grinnemo, S. Baucke, and A. Brunstrom, "Tuning SCTP Failover for Carrier Grade Telephony Signaling," *Computer Networks*, August 2009.