Towards a Decision Engine for Self-Remediating Resilient Networks

Thomas Schäfer[1], Paul Smith[1], Marcus Schöller[2], Abdul J. Mohammad[3], Justin P. Rohrer[3], David Hutchison[1], and James P.G. Sterbenz[1,3]

[1] Computing Department, Lancaster University, UK
t.schaefer1.@lancaster.ac.uk
{p.smith, dh, jpgs}@comp.lancs.ac.uk
[2] NEC Laboratories Europe, Heidelberg, Germany
marcus.schoeller@nw.neclab.eu
[3] Department of Electrical Engineering and Computer Science
University of Kansas, USA
{jabbar, rohrej, jpgs}@ittc.ku.edu

A significant aim of future networks is to make them resilient to challenges to normal operation, such as high mobility, poor wireless connectivity, and network attacks. A resilient network must autonomously defend itself, detect any challenges to the system, remediate to preserve a demanded level of service, and recover after the end of a challenge. There may be a range of remediation mechanisms available to a resilient network, which may operate at different layers of the protocol stack, and have different costs and levels of effectiveness associated with them. Therefore, a key problem in this area is to select from a set of available remediation mechanisms a suitable subset that can be used to form an effective response to address a challenge. Our research focuses on a decision engine to address this problem.

The four main entities that we believe are necessary to effectively deduce an appropriate remediation strategy from a set of available mechanisms are highlighted in Fig. 1 – we briefly describe them.

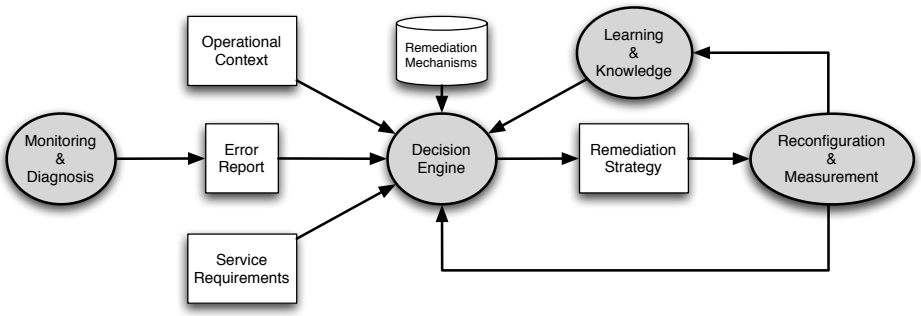

Fig. 1. The entities in a self-remediating resilient network

Symptoms of a challenge to the system (e.g., a degradation in service because of a DDoS attack) are detected by sensors as part of a constant *monitoring* phase. Upon detection of an unacceptable degradation in service, a *diagnosis* engine aims to determine what the challenge to the system is, and produces an error report. The *decision engine* aims to select an appropriate remediation strategy to mitigate the challenge to the system using the following information:

- *Error Report*
  This is a result of the diagnosis entity and describes the problem that caused an unacceptable degradation in service to occur. We envisage that this information could be incomplete, delayed, and contradictory.
- *Service Requirements*
  A description of the QoS requirements, including resilience demands, of the services that are deployed on the network. This will include the requirements of the services effected by the challenge, and those that a remediation strategy may effect (if this can be determined). Service requirements will be used when measuring the suitability of a selected strategy.
- *Operational Context*
  This is a description of the current network context, such as underlying technologies in use, network size, and administrative policies.
- *Remediation Mechanisms*
  A set of mechanisms that could be used to form a strategy to remediate against a diagnosed challenge. To aid selection, we propose to associate metadata with each mechanism that describes the challenges they can be used to remediate against, their associated costs, and arguments when used, for example.

When an appropriate strategy has been selected by the decision engine, the network must be *reconfigured* to implement it. *Measurements* must be taken to determine how effective a remediation strategy has been, if it is ineffective (or even makes the situation worse by some measure), a different approach must be selected by the decision engine. Finally, a second control loop *learns* from past events and builds a *knowledge-base*. A simple form of learning could include less readily selecting strategies that were ineffective in previous iterations of the system.

A significant challenge is to avoid the system flapping because of contradicting mechanisms and service requirements. Another problem is understanding when the end of a challenge has occurred and discontinuing a selected strategy in a safe manner. Due to the complex nature of this problem we will start our research with a centralized decision engine that, given a perfect error report and a set of service requirements, can select an appropriate strategy. We will then increase the complexity of the problem by considering imperfect error information, unavailable mechanisms in certain contexts (e.g., because of policy constraints), and decentralized autonomous decision making engines that share information.

Acknowledgements

Paul Smith's research is kindly supported by Telekom Austria AG. We gratefully acknowledge the support of the EC-funded SAC ANA Project (FP6-IST-27489) and the US NSF FIND PostModern Project.