FAULT LOCALIZATION AND SELF-HEALING WITH DYNAMIC DOMAIN CONFIGURATION

L. Kant, A. McAuley, R. Morera,
{lkant, mcauley, morera}@research.telcordia.com
Telcordia Technologies, Inc.
445 South Street, Morristown, NJ 07960, USA

A.S. Sethi, M. Steinder
{sethi, steinder}@mail.eecis.udel.edu
University of Delaware,
Newark, DE, 19716. USA

ABSTRACT

Compared to their commercial counterparts, future battlefield networks require much more extensive fault management and automation mechanisms. Much work has been done to improve these functions, since survivability and automation are seen as critical to the Army's next generation tactical and strategic battlefield networks such as FCS; however, they are generally treated separately. This paper[1] highlights the synergy between these functions. In particular we show that by reconfiguring domains, as fault localization and multi-layer self-healing algorithms mandate, we can help speed the process in isolating the cause of many of these alarms and help solve some of the network problems.

1 INTRODUCTION

The Army's next generation tactical and strategic battlefield networks are envisioned to offer a highly automated, survivable, secure and novel paradigm of battlefield operations. This paper discusses the design of two important capabilities desired from these networks, namely, **survivability and automation**.

The sporadic and hostile nature of the battlefield environment, coupled with the random/unpredictable mobility patterns of the network elements (nodes, links), result in the critical need for novel fault management mechanisms and autoconfiguration protocols. There have recently been some innovative designs for these functions in next generation tactical and strategic battlefield networks. In particular:

1) Fault Management (**FM**) Mechanisms consisting of:
   - Fault localization techniques that can perform rapid (no longer exhibit order exponential complexity) and accurate (high detection and low false positive rates) fault localization/co-relation to pinpoint the cause of an underlying failure/symptom.
   - Dynamic policy-based multi-layer self-healing mechanisms that can provide automated recovery of the various applications (esp. mission critical applications). Concentrate on techniques that are low cost and complexity for battlefield use.
2) Dynamic Domain Configuration (**DDC**) mechanisms:
   - Dynamically reconfiguring the network based on the networks' dynamics and specified policies. In particular it can dynamically create separate routing, configuration and other functional domains.

Section 2 provides a brief overview of fault localization techniques, policy-based multi-layer self-healing mechanisms, and DDC mechanisms that show promise for future battlefield networks. While recognizing that each of the components is critical, this paper proposes that an appropriate combination of the above components have the potential of yielding a much more powerful synergy than they would if treated in isolation. Examples of the synergy include:

- Fault localization can constructively use the DDC to partition the network size in order to obtain results that converge quickly that in turn increase the detection rates and reduce false positive rates (as discussed in Section 3.1).
- DDC may be invoked by the self-healing mechanism to isolate/contain misbehaving network elements and/or segments (as discussed in Section 3.2).

2 FAULT MANAGEMENT AND DCC IN BATTLEFIELD NETWORKS

This section reviews recent advances in fault localization, self-healing and dynamic domain autoconfiguration for large dynamic networks.

Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance (CTA) Program, Cooperative Agreement DAAD19-2-01-0011. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

2.1 Fault Localization

FCS networks possess many unique characteristics because of which most existing fault diagnosis techniques cannot be directly used[Steinder03a]. In the dynamic environment of mobile ad-hoc battlefield (FCS) networks, fault diagnosis must be able to deal with multiple simultaneous faults, loss of symptoms, and noise in observed data; it also must be able to diagnose availability and QoS-related problems in addition to hardware failures. Recent research shows a research methodology for fault diagnosis in FCS networks that possess the desired characteristics. It uses a multi-layer model that uses Bayesian techniques [Heckerman95, Pearl88] to capture the dependencies that may exist between entities in multiple network nodes and in multiple protocol layers at those nodes. A Bayesian algorithm operates on this model to perform fault localization[Steinder02a, Steinder02b]. Another novel algorithm processes observed symptoms incrementally and produce updates to the set of most likely hypotheses that explain the symptoms [Kant02,Steinder01,Steinder03b].

The first algorithm uses iterative belief updating [Pearl88] for singly-connected belief networks to calculate the most probable explanation of observed symptoms based on a fault propagation model with undirected loops. The algorithm processes symptoms in an event-driven manner running one iteration of belief updating for every observed symptom.

The second algorithm, called incremental hypothesis updating, creates a set of most likely hypotheses, where each hypothesis is a conjunction of faults that explains all observed symptoms. The algorithm proceeds in an event-driven and incremental fashion and ranks hypotheses using a belief metric. When a new symptom is observed, the set of hypotheses is updated with the explanation of the new symptom. If a hypothesis is unable to explain the new symptom, it is either removed from the set or is extended by adding a fault that can explain the symptom. Faults are added using a greedy heuristic that helps to limit the complexity of the algorithm.

A more detailed description of both algorithms as well as simulation results comparing their performance with other algorithms may be found in [Steinder01,Steinder 02a, Steinder 02b,Steinder03a,Steinder03].

2.2 Policy-based Dynamic Multi-layer Self-healing Mechanisms

Traditional self-healing uses redundant equipment-based single layer (physical layer – **L1**) philosophy. While L1 automatic protection switching (**APS**) approach provides low restoration delays (~50msecs), it suffers from severe handicaps in the context of battlefield networks due to the fact that it is (a) *very* resource expensive and (ii) limited to handling hard failures (i.e., equipment failures) alone. In battlefield networks, however, a substantial amount of failures are likely to be "soft" failures, i.e., failures that result from the stochastic nature of the network (e.g., excessive performance degradation). Furthermore, due to the diverse survivability requirements, it may even be useless to provide a uniform degree of restoration (i.e., delays < 50 msecs) to all of the applications. For example, while mission critical and delay sensitive applications may require stringent restoration delay guarantees, loss sensitive applications, such as battlefield terrain information may be okay with delayed albeit guaranteed restoration.

A new policy-based multi-layer self-healing mechanism [Kant02] moves restoration to higher layers of the protocol stack: in particular to the link (L2) and network (L3) layers. Examples include use of power-control at L2 or, on-demand survivability-based re-routing at L3, to achieve service survivability Thus, restoration is no longer only APS based - i.e., no longer restricted to L1 alone.

Other key aspects of our self-healing approach are (ii) the use of more than one layer and (iii) use of dynamic policy-based NM techniques to provide the required self-healing.

In addition, the Network Management Layer (NML) interacts with the Service Management Layer (SML) to obtain a list of the affected applications and their survivability requirements. Thus for example, high priority applications may be restored first and subsequently other applications may also be restored using the rules/policies that define priorities based on the survivability requirements.

Note that L1 restoration is still exploited but with the following important deviations from the traditionally used L1 self-healing. (i) The system contains very limited redundancy by having 1:N N>>1, i.e., by requiring just one dedicated resource for N working resources (vs. the traditionally used 1:1; with N=1). (ii) Since this is a multi-layer mechanism, the self-healing mechanism within the FM sub-system will pick out only a sub-set of applications to be restored at L1 - which can be policy driven (see also discussion at the end of this sub-section) and be made to correspond to the sub-set of mission critical applications. (iii) While L1 self-healing has traditionally been the default, it uses L3 as the default. The limited L1 will only be triggered if the FM subsystem determines that the set of high-priority applications cannot all be restored with low delays at L3

This multi-layer self-healing strategy lends itself excellently to the rapidly emerging policy-based network management paradigm because the self-healing

alternatives are indeed expressed as well-defined policies. Example policies include rules defining the choice of a particular restoration layer for a given set of applications, rules governing the restoration priorities of the applications at any given layer, etc. In fact, meta-rules that check for consistency may also be defined as policies, for example, policies that prevent simultaneous restoration of a given application at more than one layer, which constitutes a very important policy set. Finally, in light of stringent security requirements in an FCS environment, the proposed policy-based self-healing mechanism provides the added advantage of being able to integrate with a Security Management (SM) sub-system. It also provides the much-needed integration of Fault Management (FM), Configuration Management (CM) and Performance Management (PM). Note also that the policies/rules themselves may be static or dynamic and that the design is not restricted to the use of just static policy engines[2].

2.3 Dynamic Domain Configuration (DDC)

FCS networks may encompass a large number (e.g., 10,000) of rapidly deployed nodes with heterogeneous characteristics and capabilities. The communication links will also have vastly different speed, range and error rate characteristics. Most networking protocols, however, are suited only to particular node and link characteristics and scale only up to a maximum number of nodes. For example, routing protocol performance quickly degrades (e.g., due to slow convergence time) if nodes are "too dynamic" or the number of nodes exceeds some maximum.

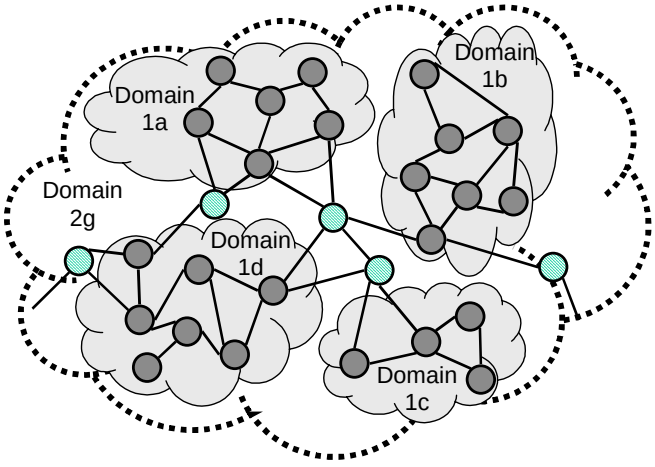

Figure 1 Example adhoc domain hierarchy

Dividing the network into independent and more homogeneous "domains" (e.g., see Figure 1) with some abstraction of intra-domain information, can help network scalability and survivability [MORERA02]. The Future Combat Systems (FCS) for example, can be automatically divided into small (e.g., 30 node) interconnected IP domains and assigning each a routing protocol that best meets the domain's characteristics. Through extensions to the IP Autoconfiguration Suite (IPAS) [MCAULEY02], the automatic creation of domains has been shown [MANOUSAKIS02].

The autoconfiguration suite is made up of four components: a) Dynamic and Rapid Configuration Protocol (DRCP), b) the Dynamic Configuration Distribution Protocol (DCDP), c) the Adaptive Configuration Agent (ACA), and the d) Configuration Reporting Protocol (YAP). DRCP extends DHCP for wireless and mobile environment. DCDP distributes configuration information to all hosts and routers in the network. The ACA provides the intelligence to optimize network configuration, allowing complex policy –based rules to be run against the latest network information (provided through YAP).

3 FAULT MANAGEMENT AND DYNAMIC DOMAIN CONFIGURATION

This section describes the novel use of DDC to perform Fault Localization and Self-healing.

3.1 Using DDC for Fault Localization

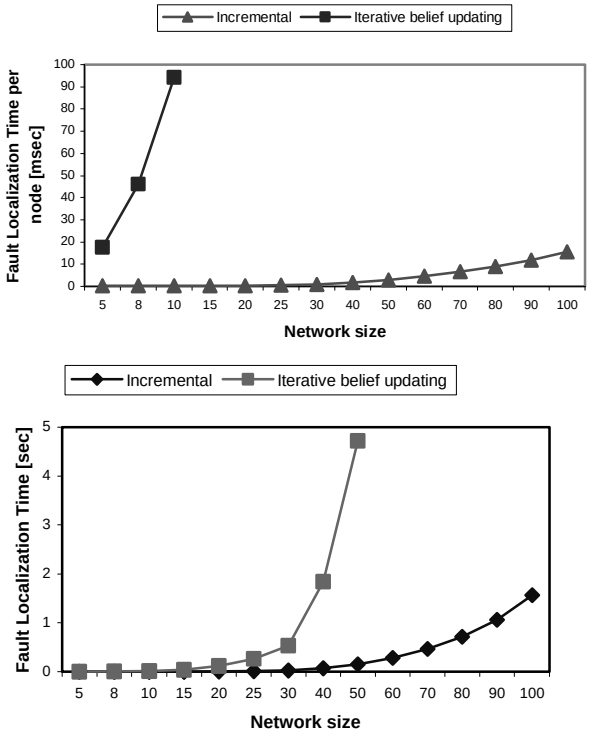

Figure 2 Fault Localization time versus Network Size

[2] However we note the design of dynamic policy engines may in turn be a challenging issue.

The fault localization algorithms described in Section 2.1 can constructively employ the DDC scheme of Section 2.3. This is achieved by deploying an instantiation of the fault correlation system in each of the network domains, and also deploying another instance of the fault management system across the domains to provide a higher layer of communication. Modifying the fault localization schemes to work in a distributed manner across multiple domains is a significant research problem that we plan to address in the near future.

Figure 2 shows how the node fault localization time and the overall fault localization time clearly increases rapidly for larger networks, even for the more efficient incremental algorithm. By exploiting the smaller sizes of the network domains, the fault correlation can complete in less time compared to operating on the whole network.

These figures do not include the overhead of communication across multiple layers (we are at present engaged in a more comprehensive study of a distributed fault correlation system to be used across domains, from which we hope to obtain more definitive results). Nevertheless, the performance benefits of dynamic domain configuration for fault management are obvious.

Additionally, the use of DDC to dynamically partition the network and therefore contain its size, may allow the use of Bayesian Belief updating-based algorithms. This is desirable because of its excellent performance (low false positives and high detection rates) [Kant02]. The smaller networks make the prohibitively high convergence times with growing network sizes, less of an issue.

3.2 Using DDC for Self-healing

This sub-section discusses the use of DDC to achieve self-healing in the event of both soft (stochastic performance related) and hard (deterministic) failures. In general, we assume that DDC is one of many alternatives for self-healing. We will describe the use of DDC through several examples:

- **Using DDC to completely isolate misbehaving nodes**. Assume that we have found several routers injecting bad routes, but they will not correct themselves or be silent. The self-healing process can inform the DDC to isolate these nodes into their own domain and not allow border nodes to pass information from that "bad" domain into the rest of the system.
- **Using DDC to partially isolate misbehaving nodes**. Assume some nodes are injecting frequent route updates into the network causing congestion or the non-converge of routes. Assume also that the mission goals dictate that the self-healing process must maintain some communication with these nodes. If the DDC puts these "flapping nodes" into a separate domain, the routing problems can be significantly reduced or even eliminated; yet, it is still possible to send and receive packets from these nodes, through the border nodes (which still advertise the routes).
- **Using DDC to fix service problems nodes**. Assume that nodes need to use some servers (e.g., for DNS or SIP). Initially the network may function well with a single server and applications get their desired performance. However, if the links to the server start to deteriorate or the load on the server increase, the applications could start getting into performance problems. The self-healing could tell DDC to split the server domain into, creating new servers; or the DCC could simply move the server functionality onto a better node.
- **Using DDC to isolate intermittent links**. Assume a group of soldiers involved in an "exploring mission." Initially, all the nodes connect using a wireless ad-hoc network with high quality links and little traffic among the nodes. Then, the DCC configures a single domain, with all nodes running a MANET routing protocol such as AODV. As the mission evolves, a small group of soldiers move into a region with very poor radio link quality and high link failures. The routing protocol is not able to cope with such network dynamics and communication between nodes becomes very difficult. The self-healing then uses the DDC to send a command to reconfigure the network and split the original routing domain in two routing domains, one still running the original routing protocol and the other using simple flooding. This way, nodes in the more stable region are not affected by the unstable links in the dynamic region.

Finally, before we conclude this sub-section, we observe the following cost-, performance-, and complexity-sensitivity aspects of our approach. Our proposed multi-layer restoration mechanism is a judicious and novel combination of self-healing at different levels and has the merits of providing cost and performance sensitive restoration. The reductions in cost are obvious since L3 self-healing, such as DCC, is essentially non-redundant. When L3 self-healing is not possible, we can still invoke some L1 restorations (e.g., inject an airborne or other router) with very little redundancy. The performance sensitivity is achieved by understanding that not all of the applications will require the same degree of survivability and hence tailoring the restoration of high-priority applications first followed by the others as also indicated in our studies in [Kant02].

The complexity-sensitivity aspect of our multi-layer approach is achieved due to the fact that our self-healing/service survivability mechanisms are designed to work both directly and indirectly, with a majority of existing performance management (PM) and configuration management (CM) functionalities. For example, responding to and analyzing performance-related alarms (caused by "soft failures") and responding to soft failures by re-routing, re-configuring essentially imply close tie-in with existing PM, CM and/or routing functionalities, albeit with some modifications.

3.3 Linking DDC and Fault Management

In a distributed architecture, where the fault and configuration management are two different subsystems, the ACA and the policy management agent for the FM must cooperate in order to use DDC in FM in the way described in the previous section. In a more centralized architecture, both functionalities (configuration and fault management) may be integrated in the ACA.

Additionally, the FM and the CM subsystems rely on the network state and configuration information, (e,g number of nodes in a domain, routing protocol, domain type, border nodes, servers …) stored in databases to properly apply their policies. The configuration reporting protocol (YAP) is to be enhanced to collect and report network information to be used by the FM as well as CM in IPAS.

4 SUMMARY AND CONTINUING WORK

This paper highlights a powerful synergy between various battlefield functionalities. In particular, the inter-dependencies between dynamic autoconfiguration mechanisms to provide automation (being done in ARL CTA C&N Task 1.2) and fault management operations such as fault localization and self-healing mechanisms to provide survivability (ARL CTA C&N Task 1.4). The paper outlines an approach that combines the above in order to be able to realize novel and advanced battlefield operations in the Army's next generation of tactical and strategic networks.

Continuing work in the area of fault localization and DDC includes study of a distributed fault correlation system that can be used across domains. In the area of self-healing with DDC, future work involves the investigation of overheads and a quantitative analysis of the benefits.

5 ACKNOWLEDGEMENTS

The authors are grateful to the ARL CTA C&N program for funding the work described in this paper.

6 BIBLIOGRAPHY

[Heckerman95] D. Heckerman and M.P. Wellman, "Bayesian networks." *Communications of the ACM*, 38(3): 27-30, Mar. 1995.

[Kant02] L. Kant, A.S. Sethi, and M. Steinder, "Fault Localization and Self-Healing Mechanisms for FCS Networks." In *Proc. 23rd Annual Army Science Conference*, Orlando, FL, 2002.

[Manousakis02] K. Manousakis, A. McAuley, R. Morera, J. Baras, "Routing Domain Autoconfiguration for More Efficient and Rapidly Deployable Mobile Networks", Army Science Conference, Dec. 2002.

[Mcauley02] A. McAuley, D. Chee, J. Chiang, S. Das, K. Manousakis, R. Morera, L. Wong, K. Young, "Automatic Configuration and Reconfiguration in Dynamic Networks," Army Science Conference, December 2002.

[Morera02] R. Morera, A. McAuley, "Flexible Domain Configuration for More Scalable, Efficient and Robust Battlefield Networks" IEEE MILCOM, October 2002.

[Pearl88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, 1988.

[Steinder01] M.Steinder and A.S.Sethi, "Non-deterministic diagnosis of end-to-end service failures in a multi-layer communication system". In Proc. Of ICCCN, pp. 374-379, Scottsdale, AZ, 2001

[Steinder02a] M. Steinder and A.S. Sethi, "End-to-end service failure diagnosis using belief networks." In *Proc. of Network Operations and Management Symposium (NOMS)*, Florence, Italy, 2002.

[Steinder02b] M. Steinder and A.S. Sethi, "Increasing robustness of fault localization through analysis of lost, spurious, and positive symptoms." In *Proc. of IEEE Infocom*, New York, NY, 2002.

[Steinder03b] M. Steinder and A.S. Sethi, "Application of Bayesian Reasoning Techniques to Fault Localization in FCS Networks", Submitted to *CTA Annual Symposium*, 2003. [i]

---

[i] The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of the Army Research Laboratory or the U.S. Government