International Research Journal of Computer Science (IRJCS)
Issue 04, Volume 3 (April 2016)
ISSN: 2393-9842
www.irjcs.com

UNSTURUCTERED FAULT NODE DETECTION AND NEUTRALIZING IN OVERLAY NETWORK

Thara.P
Research Scholar
Mazharul Uloom College
Ambur – 635 802

Joseph Gabriel.S
Head & Associate Professor/ CS
Mazharul Uloom College
Ambur – 635 802

Rizwan Ahmed P
Head & Asst. Professor /CA
Mazharul Uloom College
Ambur – 635 802

Abstract— This project data are automatically managed and corrected in efficient manner. We are interested in diagnosing and repairing faulty nodes in a managed overlay network, in which overlay nodes are independently operated by multiple administrative domains. By an administrative domain, we mean a single administrative authority that controls a collection of resources. Successful data delivery is preserved with a very high probability even if a subset of overlay nodes is failed. While data may be re-routed to bypass failed nodes, robustness of data delivery will be degraded if the failed nodes are not immediately repaired because attackers can now devote resources to attack the remaining non-failed nodes. We are implemented security-oriented overlay networks might be deployed over a number of end sites rather than a single ISP and can be viewed as externally managed networks since each end site is independently operated. Thus, our proposed network fault correction mechanism is also essential for this type of networks. We propose to infer the most likely faulty IP links with the assumption of significantly small failure probabilities. All these schemes focus on localizing faults. In this paper, in addition to probabilistic fault localization, we take the subsequent step of repairing faults as well. We show that checking first the most likely faults does not give optimal results in general. Our work considers the case where nodes are externally managed and they need to be manually checked in order to determine failures. Fault correction has also been studied extensively. Direct access and monitor nodes that are independently operated by different administrative domains, but instead we must infer failures via end to- end measurements.

Index Terms — Network management, network diagnosis and correction, fault localization and repair, reliability engineering

INTRODUCTION

Network components are prone to a variety of faults such as packet loss, link cut, or node outage. To prevent the faulty components from hindering network applications, it is important to *diagnose* (i.e., *detect* and *localize*) the components that are the root cause of network faults. However, it is also desirable to *repair* the faulty components to enable them to return to their operational states. Therefore, we focus on *network fault correction*, by which we mean not only to diagnose, but also to repair all faulty components within a network. In addition, it has been shown that a network outage can bring significant economic loss. For example, the revenue loss due to a 24-hour outage of a Switzerland-based Internet service provider can be more than CHF 30 million. As a result, we want to devise a *cost-effective* network fault correction mechanism that corrects all network faults at minimum cost.To diagnose (but not repair) network faults, recent approaches use all network nodes to collaboratively achieve this. For instance, in hop-by-hop authentication, each hop inspects packets received from its previous hop and reports errors when packets are found to be corrupted. While such a distributed infrastructure can accurately pinpoint network faults, deploying and maintaining numerous monitoring points in a large-scale network introduces heavy computational overhead in collecting network statistics and involves complicated administrative management. In particular, it is difficult to directly monitor and access all overlay nodes in an *externally managed network*, whose routing nodes are independently operated by various administrative domains. In this case, we can only infer the network condition from end-to-end information.

Here, we consider an *end-to-end* inference approach which, using end-to-end measurements, *infers* components that are probably faulty in forwarding data in an application-layer overlay network whose overlay nodes are externally managed by independent administrative domains. We start with a routing tree topology with a set of overlay nodes, since a tree-based setting is typically seen in destination-based routing, where each overlay node builds a routing tree with itself as a root, as well as in multicast routing, where a routing tree is built to connect members in a multicast group. We then monitor every root-to-leaf overlay path. If a path exhibits any “anomalous behavior” in forwarding data, then some “faulty” overlay node on the path must be responsible. In practice, the precise definition of an “anomalous behavior” depends on specific applications. For instance, a path is said to be anomalous if it fails to deliver a number of correct packets within a time window. Using the path information collected at the application endpoints (i.e., leaf nodes), we can narrow down the space of possibly faulty overlay nodes. we cannot directly monitor and access externally managed overlay nodes, in order to correct the faulty nodes, we need to contact the administrators of the corresponding domains to manually check a sequence of potentially faulty nodes and fix any nodes that are found to be actually faulty.

Given the anomalous paths in a tree, our main goal is to infer the best node (or the best set of nodes) that should be first checked so as to minimize the expected cost of correcting all faulty nodes.

In this paper, we develop several optimality results for inferring the best node that should be first checked by a network fault correction scheme, with an objective to minimize the expected cost of correcting all faulty nodes. Our contributions include the following:

- *Conventional failure localization problems seek to identify the most likely faulty node (i.e., the node with the highest conditional failure probability given a network with faulty nodes). Here, we show that first checking the node that is most likely faulty or has the least checking cost does not necessarily minimize the expected cost of correcting all faulty nodes.*
- *We formally identify a subset of nodes termed candidate nodes, one of which should be first checked in order to minimize the expected cost of correcting all faulty nodes. We develop a potential function that determines the candidate nodes.*
- *Based on the potential function and candidate nodes that we propose, we devise various heuristics for the best node inference in a single tree and multiple trees, where the latter forms a more general topology. We show via simulation that the candidate node with the highest potential value is in fact the best node that should be first checked by an optimal strategy in at least 95% of time under a special setting. In addition, we conduct simulation using large-scale network topologies. As compared to the strategies that first check the most likely faulty node (or the set of most likely faulty nodes), we show that by first checking the candidate nodes, we can decrease the checking cost of correcting all faulty nodes, for example, by more than 30% in some scenarios.*

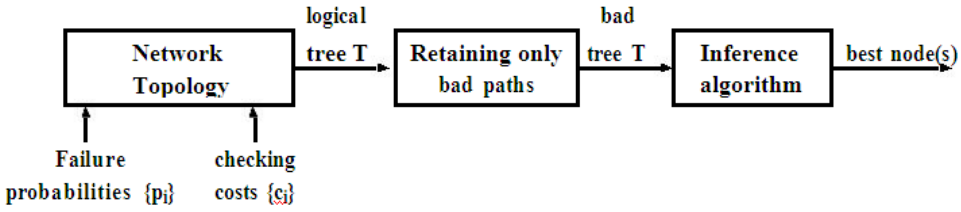

Fig. 1. End-to-end inference approach for a network fault correction scheme.

EXTERNALLY MANAGED OVERLAYNETWORKS

In this paper, we are interested in diagnosing and repairing faulty nodes in an *externally managed overlay network*, in which overlay nodes are independently operated by multiple *administrative domains*. By an administrative domain, we mean a single administrative authority that controls a collection of resources (e.g., routers and servers). Examples of externally managed overlay networks include Resilient Overlay Network (RON), which provides routing resilience toward Internet path outages, and Service Overlay Network (SON), which provides end-to-end quality-of-service guarantees. Both RON and SON deploy overlay nodes over multiple administrative domains that cooperatively accomplish certain network services. To ensure the availability of these network services, an effective network fault mechanism is therefore necessary.

Researchers also advocate the notion of security-oriented overlay architectures, such as Secure Overlay Services (SOS) and Mayday, to defend against denial-of-service attacks. In both SOS and Mayday, data is securely tunneled over an overlay network. Successful data delivery is preserved with a very high probability even if a subset of overlay nodes are failed (e.g., shut down by attackers). While data may be re-routed to bypass failed nodes, robustness of data delivery will be degraded if the failed nodes are not immediately repaired because attackers can now devote resources to attacking the remaining non-failed nodes. Note that security-oriented overlay networks might be deployed over a number of end sites rather than a single ISP and can be viewed as externally managed networks since each end site is independently operated. Thus, our proposed network fault correction mechanism is also essential for this type of networks.

PROBLEM FORMULATION

In this section, we formulate our end-to-end inference approach for network fault correction. We are interested in diagnosing and repairing faulty nodes in an externally managed overlay network, in which the overlay nodes cannot be directly accessed. In order to diagnose failures, we need to contact the administrators of the corresponding domains to manually *check* potentially faulty nodes (e.g., by conducting a set of sanity tests). Any nodes that are found to be actually faulty will then be repaired.

Figure 1 summarizes the end-to-end inference approach. We consider a logical tree as $T = (N, \{p_i\}, \{c_i\})$, where N is the set of overlay nodes, p_i is the failure probability of node $i \in N$, and c_i is the checking cost of deciding if node $i \in N$ is faulty.

The overlay node set N provides the topological information and specifies the sequence of overlay nodes, and hence the corresponding administrative domains along which each data packet are traversed. On the other hand, the construction of $\{p_i\}$ and $\{c_i\}$are based on the following failure and cost models, respectively.

Failure model: Our analysis focuses on overlay node failures such that nodes cannot properly forward data. For example, nodes can delay or drop packets to be forwarded because of power outage, full transmission queues, hardware errors, or route misconfiguration. In addition, our analysis focuses on *fail-stop* failures, meaning that a node completely stops its operations upon failures. Examples of fail-stop failures include power outage or machine shutdown. Under the fail-stop model, the failure probabilities $\{p_i\}$ can then be characterized via statistical measurements of reliability indexes or vulnerability modeling . However, we note that constructing accurate failure probabilities for overlay nodes is difficult in practice. Thus, we also evaluate the impact on our proposed approach due to inaccurate estimates of failure probabilities. Here, we consider the case where each overlay node is a physical node that possesses a single identity, and hence we assume that node failures and the corresponding failure probabilities $\{p_i\}$ are all independent.

Cost model: We characterize the checking costs $\{c_i\}$ using the personnel hours and wages required for troubleshooting problems or the cost of test equipment. Thus, the checking costs can be highly varying, depending on the administrative domains in which the checked nodes resides. Note that we do not consider the cost of repairing overlay nodes, and that the total checking cost is the only cost component being considered in our optimization problem.

In our analysis, we assume that p_i and c_i can be any values in [0, 1] and [0,∞], respectively. For instance, if $c_i = 1$ for all i, then the total checking cost denotes the number of nodes that have been checked. Also, depending on the fault definition, the failure probabilities $\{p_i\}$ can be significantly small for general failures , but can also be non-negligible if we are concerned with how likely a node is brought down due to catastrophic events that lead to large-scale failures.

Each node in a logical tree T is classified as faulty or non-faulty, depending on how we first define whether a (root-to-leaf) path exhibits any "anomalous behavior". In practice, such a definition varies across applications. For example, we say that a path behaves anomalously if it fails to deliver a number of correct packets within a time window, and that some node on the path is faulty if it causes severe packet loss and delay. Note that the inference approach only knows whether a path is anomalous, but does not specifically know which and how many nodes on the path are faulty.

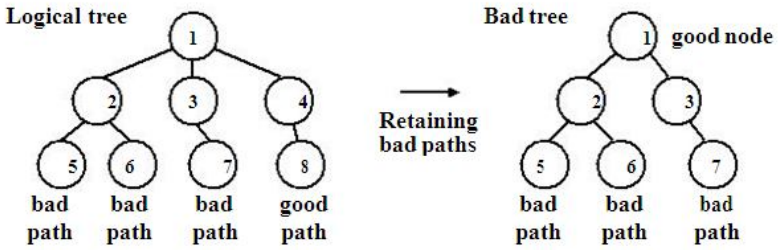

Fig. 2- Given a logical tree, we retain only the bad paths and indicate any good node. Since path (1,4,8) is a good path, it is known that nodes 1, 4, and 8 are good. Nodes 4 and 8 can be pruned from the tree, and node 1 can be indicated as good. The resulting set of bad paths will lead to a bad tree.

To help our discussion, each node in T is referred to as *bad* if it is faulty, or as *good* otherwise. We say a path is *bad* if it contains at least one bad node, and is otherwise *good*. Also, we call T a *bad tree* if every path in the tree is a bad path.Since our failure model focuses on fail-stop failures, we further assume that a node exhibits the same behavior across all paths upon which it lies. With this assumption, if a node lies on at least one good path, then we infer that it is a good node. Note that a good node may still lie on one or more bad paths, but this only means each such bad path contains some other bad node. On the other hand, if a node exhibits different behaviors across different paths, our analysis becomes more complicated, as a good path now possibly contains bad nodes as well. We pose the analysis of different behaviors of a node as future work.

Given a logical tree T, we determine whether a path is good or bad via end-to-end measurements that are carried out between the root and leaf nodes of T . For example, the root can send *probes* to the leaf nodes, from which we collect the measurement results. Since we focus on the data-forwarding failures, the probes should represent the regular data packets that can be forwarded by overlay nodes. Since a good path contains only good nodes that need not be checked, we only need to focus on the bad paths in T. Also, any node that lies on both good and bad paths is indicated to be good. To illustrate, Figure 2 shows how to retain only the bad paths in T and indicate the good nodes. The resulting set of bad paths will then form a bad tree. With a slight abuse of notation, we denote this bad tree by T as well.

We then pass the bad tree T to the *inference algorithm*, which determines, from the set of nodes that are not indicated as good, the "best" node (or the "best" set of nodes) to be checked, and repaired if necessary. Before formalizing the notion of "best", we first state our optimization goal, namely, to minimize the expected cost of correcting all faulty nodes in a given bad tree T . Here, we assume that using end-to-end measurements (i.e., sending probe packets from the root to leaf nodes) to determine bad paths incurs negligible cost. Thus, the correction cost has two main components: the cost of checking all nodes and the cost of repairing all faulty nodes. However, we do not consider the repair cost since all faulty nodes have to be recovered eventually, and any successful repair strategy has the same total cost of repairing all faulty nodes. As a result, by cost, we here refer to the checking cost only.

Because of our optimization goal, we can focus on the *sequential* case where we check one node at a time, since checking multiple nodes simultaneously does not improve the expected checking cost, even though it reduces the time required to repair all bad nodes on all bad paths. As a result, our theoretical analysis assumes that the inference algorithm returns only a *single* best node, while we evaluate via simulation the impact of simultaneously inferring and checking multiple nodes. With the optimization goal, we select the "best" node based on a *diagnosis sequence* $S = (l_1, l_2,\ldots l_{|N|})$, defined as the order of nodes to be examined given a bad tree T . When node l_i is examined, it is either *checked* or *skipped*. If node l_i lies on bad paths only, we cannot tell whether it is good or bad. In this case, we have to check node l_i, and repair it if it is determined to be bad. On the other hand, if node l_i lies on a good path, it is known to be a good node and does not need to be checked. In this case, we say we skip node l_i. After node l_i has been checked or skipped, it is known to be good.

A diagnosis sequence S is said to be *optimal* if its expected cost is minimum among all possible diagnosis sequences. Therefore, among all the nodes in a bad tree T , we formally define the *best node* as the first node in an optimal diagnosis sequence for T . In other words, the best node should be the node to be first checked in order to minimize the expected cost of correcting all faulty nodes.We point out that the optimal decision of the inference algorithm is derived from the current topology. The decision will be revised for a new topology when the faults are actually checked and repaired. In spite of this, the topology may still change between the time of identifying the physical topology and performing the inference algorithm. However, topology change occurs in a coarse time scale (on the order of minutes and hours). Thus, as long as the network fault correction scheme has its monitoring period bounded within the time scale of topology changes, the topology should remain fairly stable.

We seek to answer the following question: Given a bad tree T , how can the inference algorithm determine the best node in polynomial time?

Algorithm 1 *Brute-force inference algorithm*

Input: Bad tree $T = (N, \{p_i\},\{c_i\})$
1: $S^* = \emptyset$, $c^* = \infty$
2: ***for all*** *diagnosis sequence S* ***do***
3: compute c = the expected cost of S
4: ***if*** $c < c^*$ ***then*** $S^* = S$, $c^* = c$
5: return the first node in S^*

CANDIDATE NODES

Instead of the naive choices described in the previous section, we show in this section that we should first check a *candidate node*, which is selected based on the maximization of a *potential function* as described below.We first give the notation and definitions that we will use. Given a tree T , we define *ancestors* of node i to be the nodes (not including node i) on the path from the root of T to node i, and *descendants* of node i to be the nodes that have node i as one of their ancestors. Let T be the event that T is a bad tree, and X_i be the event that node i is a bad node. Let A_i be the event that the ancestors of node i are all good. If node r is the root node, then we let A_r be always true and $\Pr(A_r) = 1$.

RELATED WORK

Our end-to-end inference solution is also use end-to-end measurements to infer link statistics or the underlying network topology. We extend this end-to-end approach to the application of network fault management. Fault diagnosis is an important topic in network management. For example, the codebook approach and the Bayesian approach consider a deterministic setting where network faults are equally likely to occur. To address probabilistic faults, to localize the most probable subset of faults given the observed symptoms. Katzela and Schwartz show that such a problem is generally NP-hard. In view of this, Steinder and Sethi formulate the problem based on bipartite belief networks and propose efficient techniques on localizing end-to-end multi-layer failures. Kandula *et al.* propose to infer the most likely faulty IP links with the assumption of significantly small failure probabilities. All these schemes focus on localizing faults. In this paper, in addition to probabilistic fault localization, we take the subsequent step of repairing faults as well.

We show that checking first the most likely faults does not give optimal results in general. Consider various dimensions of performance of failure detection in overlay networks, in which nodes periodically probe their neighbors to determine failures. Our work considers the case where nodes are externally managed and they need to be manually checked in order to determine failures. Fault correction has also been studied extensively in reliability engineering, whose objective is also to minimize the cost of correcting all faulty units. However, the optimality assumes a *series* system, which is equivalent to a single path in a network setting. In contrast, we consider a routing tree whose paths are shared and dependent.

CONCLUSIONS

We present the optimality results for an end-to-end inference approach to correct (i.e., diagnose and repair) probabilistic network faults at minimum expected cost. One motivating application of using this end-to-end inference approach is an externally managed overlay network, where we cannot directly access and monitor nodes that are independently operated by different administrative domains, but instead we must infer failures via end-to-end measurements. We show that first checking the node that is most likely faulty or has the least checking cost does not necessarily minimize the expected cost of correcting all faulty nodes. In view of this, we construct a potential function for identifying the candidate nodes, one of which should be first checked by an optimal strategy. Due to the difficulty of finding the best node from the set of candidate nodes, we propose several efficient heuristics that are suitable for correcting fault nodes in large-scale overlay networks. We show that the candidate node with the highest potential is actually the best node in at least 95% of time, and that checking first the candidate nodes can reduce the cost of correcting faulty nodes as compared to checking first the most likely faulty nodes.

REFERENCES

[1] M. Adler, T. Bu, R. Sitaraman, and D. Towsley. Tree Layout for Internal Network Characterizations in Multicast Networks. In *Proc. of NGC'01*, 2001.
[2] D. Andersen. Mayday: Distributed Filtering for Internet Services. In *4th Usenix Symposium on Internet Technologies and Systems*, Mar 2003.
[3] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Re-silient Overlay Networks. In *Proc. of the 18th ACM Symposium on Operating Systems Principles (SOSP)*, Oct 2001.
[4] I. Avramopoulos, H. Kobayashi, R. Wang, and A. Krishnamurthy. Highly Secure and Efficient Routing. In *Proc. of IEEE INFOCOM*, March 2004.
[5] A.-L. Barabasi´ and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286:509–512, Oct 1999.
[6] Y. Bejerano and R. Rastogi. Robust Monitoring of Link Delays and Faults in IP Networks. In *Proc. of IEEE INFOCOM*, 2003.
[7] R. Caceres,´ N. Duffield, J. Horowitz, and D. Towsley. Multicast-based Inference of Network-Internal Loss Characteristics. *IEEE Trans. on Information Theory*, 45(7):2462–2480, November 1999.
[8] K. Carlberg. Emergency Telecommunications Services (ETS) Re-quirements for a Single Administrative Domain, Jan 2006. RFC 4375.
[9] Clip2. The Gnutella Protocol Specification v0.4. Available on http://www.limewire.com.
[10] M. Coates, A. O. Hero, R. Nowak, and B. Yu. Internet Tomogra-phy. *IEEE Signal Processing Magazine*, pages 47–65, May 2002.
[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Intro-duction to Algorithms*. MIT Press, 2nd edition, 2001.
[12] D. Dagon, G. Gu, C. P. Lee, and W. Lee. A Taxonomy of Botnet Structures. In *Proc. of NDSS*, 2007.
[13] W. Du and A. Mathur. Testing for Software Vulnerability Using Environment Perturbation. In *Proc. of the International Conference on Dependable Systems and Networks*, 2000.
[14] Z. Duan, Z.-L. Zhang, and Y. Hou. Service Overlay Networks: SLAs, QoS, and Bandwidth Provisioning. *IEEE/ACM Trans. on Networking*, 11(6):870–883, Dec 2003.
[15] T. Dubendorfer,¨ A. Wagner, and B. Plattner. An Economic Damage Model for Large-Scale Internet Attacks. In *Proc. of IEEE WETICE*, Jun 2004.
[16] N. Duffield, J. Horowitz, F. L. Presti, and D.Towsley. Multicast Topology Inference from Measured End-to-End Loss. *IEEE Trans. on Information Theory*, 48:26–45, January 2002.
[17] B. Gnedenko and I. A. Ushakov. *Probabilistic Reliability Engineer-ing*. John Wiley & Sons, Inc., 1995.
[18] S. Kandula, D. Katabi, and J.-P. Vasseur. Shrink: A Tool for Failure Diagnosis in IP Networks. In *ACM SIGCOMM MineNet-05*, Aug 2005.
[19] I. Katzela and M. Schwartz. Schemes for Fault Identification in Communication Networks. *IEEE/ACM Trans. on Networking*, 3(6):733–764, 1995.

