2009 IEEE International Conference on Networking, Architecture, and Storage

Community-base Fault Diagnosis Using Incremental Belief Revision

Yongning Tang
Illinois State University
Normal IL, USA
ytang@ilstu.edu

Guang Cheng
SouthEast University
Nanjing, China
gcheng@njnet.edu.cn.

Zhiwei Xu
University of Michigan-Dearborn
Dearborn MI, USA
zwxu@umich.edu.

Ehab Al-Shaer
DePaul University
Chicago IL, USA
ehab@cs.depaul.edu.

***Abstract*—Overlay networks have emerged as a powerful and flexible platform for developing new disruptive network applications. The attractive characteristics of overlay networks such as planetary-scale distributions, user-level flexibility (e.g. overlay routing) and manageability bring to overlay fault diagnosis new challenges, which include inaccessible underlying network information, incomplete and inaccurate network status observations; dynamic symptom-fault causality relationships, and multi-layer complexity. To address these challenges, we propose a distributed user-level *Belief Revision* based overlay fault diagnosis technique called *EUDiag*. *EUDiag* can passively use observed overlay symptoms as reported by overlay monitoring agents to correlate and diagnose faults, and select the least-costly appropriate probing actions whenever necessary to enhance the passive fault reasoning results. *EUDiag* adapts to the changes in highly dynamic overlay networks by incrementally revising user beliefs based on new observed overlay symptoms. *EUDiag* can diagnose faults without relying on underlying network fault probabilistic quantifications (e.g. prior fault probability). Simulations and experimental studies show that *EUDiag* can efficiently (e.g. low latency) and accurately localize root causes of overlay faults/problems, even when the observed symptoms are incomplete.**

I. Introduction

Overlay service model [1] has been widely adopted by research community [3][6][22] as well as commercial Overlay Service Providers (OSPs) [2][5] as an effective approach to implement disruptive planetary-scale network applications. However, overlay applications are prone to variety of faults across multiple layers such as fiber cuts, router misconfigurations, or overlay node outages. These faults usually can be observed as certain end-to-end network disorders [4][29] (e.g. packet loss or abnormal latency) and manifested as trouble tickets or monitoring alerts. The performance and reliability of overlay applications depend on the capability of overlay networks to quickly and accurately detect and diagnose faults so as to dynamically adjust their topologies [8]. However, traditional fault management techniques [14][17][18] and proprietarily developed overlay fault detection and diagnosis approaches [8][12][26][29] can not satisfy new requirements and tackle new challenges in dynamic overlay networks.

A. New Challenges in Overlay Fault Diagnosis

We believe the following new characteristics and challenges decide overlay fault diagnosis has to adopt a new approach, which is the focus of this work:

- *Inaccessible underlying network information and incomplete network status observation:* In overlay network domain, overlay services are provisioned, operated by OSPs on the top of opaque underlying networks. Overlay fault diagnosis technique must be developed based on incomplete and insufficient user-level observations.
- *Planetary-scale and widely distributed service infrastructure:* Overlay services usually run across multiple ISPs and are widely distributed [2][3]. Multiple overlay applications (e.g. Yahoo, Amazon) may co-exist on top of the common overlay network infrastructure (e.g. Akamai).
- *Multi-layer complexity and dynamic symptom-fault causality:* In overlay networks, observed symptoms are usually not designed for monitoring specific faults (e.g. malfunctioning router interface). Symptom-fault causality relationship is dynamic and unpredictable in overlay networks.
- *Fault reasoning goal and granularity:* In overlay network, it becomes more interesting for overlay applications to effectively bypass detected faulty components instead of fixing them, which is very likely impossible (e.g. for components owned by ISPs). Thus, for overlay fault reasoning, coarse-grained fault reasoning result is acceptable and may be more preferable in order to improve fault diagnosis and recovery efficiency.

Considering the above challenges, we argue that in planetary-scale multi-layer overlay networks, overlay fault diagnosis should aim at the following objectives. They should:

- *be able to isolate multiple simultaneous faults as well as faults on multiple layers*. This feature improves the technique's applicability to planetary-scale systems; and provides critical information for taking appropriate overlay fault recovery strategies.
- *be flexible to diagnosis granularity*. Overlay applications are service-oriented. The goal of overlay fault diagnosis is to help overlay applications bypass faulty components (e.g. routers in ISPs). Thus, coarse-grained diagnosis is more preferable by considering the effectiveness and performance of overlay fault diagnosis.
- *be resilient to incomplete and spurious symptoms*. Over-

978-0-7695-3741-2/09 $25.00 © 2009 IEEE
DOI 10.1109/NAS.2009.24

lay fault diagnosis technique should be robust and effective in handling incomplete or even spurious symptoms.
- *be distributed and collaborative*. Scalability and wide distribution of overlay applications [2] decide that overlay fault diagnosis has to adopt a distributed approach collaboratively to improve the diagnosis performance, minimize network intrusiveness and tackle observation incompleteness.
- *be adaptive to new observations*. Previous diagnosis result should be incrementally revisable by considering new evidence/symptoms.
- *be efficient and reasonably accurate*. Overlay fault diagnosis accuracy should be considered as a trade-off between response time and overlay recovery effectiveness.
- *be passive and active integrated*. If passive diagnosis results are not sufficient and satisfactory in terms of their credibility, a set of optimally selected probing actions should be selected to actively discover more fault indications to facilitate overlay fault diagnosis process.

B. Our Contributions

This paper investigated an interesting and timely problem: overlay fault diagnosis, from user perspective without relying on any probability fault diagnosis model. The paper makes the following contributions to the field of overlay fault diagnosis.

- It advances the state-of-the-art in overlay fault diagnosis by providing a novel user-level overlay fault reasoning framework. To the best of our knowledge, this is the first comprehensive user-level distributed and collaborative overlay fault diagnosis framework.
- It proposes a novel concept *Belief Revision* to flexibly correlate dynamic overlay symptom-fault causality, which can also be incrementally revised when new symptoms observed.
- It seamlessly incorporates active actions into passive fault reasoning process to achieve better diagnosis performance and accuracy.
- It provides an unified framework with incremental *Belief Revision* capability for OSPs collaboratively sharing monitoring information. The experiments also show *Collaboration Gain* for OSPs by sharing network observations.

The paper is organized as the following. In Section II, we formalize the problems and overview *EUDiag* system. In Section III, we introduce the concept of *Belief Revision* and elaborate technique details of *EUDiag*. In Section IV, we present our simulation studies and real experiment results to evaluate *EUDiag* performance and accuracy. In Section VI, related work is discussed. Section VII gives our conclusions and future work.

II. SYSTEM OVERVIEW AND PROBLEM FORMALIZATION

The aim of this work is to provide a general and practical fault diagnosis framework for overlay users, developers and operators (e.g. Planet-lab [3], Akamai [2]) to collaboratively monitor and analyze overlay network status and identify faults (e.g. the overlay or underlay component which caused packet loss) that degrade the application performance. Overlay service model aims at providing planetary-scale services on the top of wide area underlying network infrastructure (e.g. the Internet).

TABLE I
OVERLAY NETWORK DISTRIBUTION

	Overlay Service Providers	
	Planet-Lab	Akamai
Nodes	∼800	∼25,000
Networks	∼400	∼1,000
Countries	∼30	∼70
Applications	CoDeeN, CoralCDN	Yahoo, Facebook

As shown in TABLE I, OSPs usually provide planetary-scale network infrastructure and host multiple long-running overlay applications. However, among OSPs and even among applications hosted by the same OSP, they usually do not share monitoring information and use their own monitoring systems, which may still rely on centralized monitoring mechanism (e.g. [4]) to aggregate widely distributed and less likely relevant events. As shown in [13], the stability of edge network is less than backbone network. Thus, proximal users' observation (based on geographical or network distribution) are more likely relevant. Thus, to tackle the above inefficiency, we propose a new overlay diagnosis framework called *EUDiag*, which adopts a distributed and collaborative approach such that it can (1) effectively deal with observation incompleteness by collaboratively sharing information; (2) efficiently aggregate relevant events/observations in a distributed fashion.

In *EUDiag* framework, there are a set of Overlay Service Providers as $O = \{OSP_1, OSP_2, \cdots, OSP_N\}$. For each OSP_i, it hosts a set of overlay applications $APP_i = \{app_{i1}, app_{i2}, \cdots, app_{iM}\}$. For each overlay application app_{ij} hosted by OSP_i, it may generate a set of symptoms $S_{ij} = \{s_{ij1}, s_{ij2}, \cdots, s_{ijQ}\}$. We assume all OSPs collaboratively host a monitoring infrastructure that consists of a set of well-distributed overlay monitoring agents (m_k) $M = \{m_1, m_2, \cdots, m_K\}$. Accordingly, each application app_{ij} may also designate K application monitoring agents (denoted as $appAgents^k_{ij}$). Here, $1 \leq i \leq N, 1 \leq j \leq M, 1 \leq k \leq K; K, M, N \geq 1$. Overlay application nodes should report observed symptoms to their proximate $appAgents^k_{ij}$, which further aggregated to m_k. There are many techniques proposed for finding proximate agents such as the one discussed in [10]. In this manner, the symptoms from same application app_{ij} are partitioned into $S_{ij} = s^1_{ij} \cup s^2_{ij} \cdots \cup s^K_{ij}$.

For each distributed overlay monitoring agent m_k, the processs of overlay fault diagnosis is to find credible hypothesis (H_k), which can best explain all received symptoms $\{s^k_{ij}\}$. In order to make the following discussion easier and clearer, we first define a few important concepts and their notations in overlay network context.

Definition 1: Overlay Network Component Set (C): C consists of *Overlay Component Set* C^o and *Underlay Component Set* C^u ($C = C^u \cup C^u$). In *EUDiag* framework, each object c

($c \in C$) is assigned an unique ID as the following: if $c \in C^o$) (e.g. overlay node), c has a 1-byte prefix equal to "1"; if $c \in C^u$ (e.g. network router), c has a 1-byte prefix equal to "0". The suffix of each object is a 128-bit output from hash function (e.g. MD5) $H(getASN(c), getNetID(c))$. Here, *getASN* and *getNetID* are two utility functions to obtain AS number (*ASN*) and corresponding network ID (*NetID*) of a given component c. In order to avoid ambiguity of Variable Length Subnet Masking (VLSM) adopted in CIDR notation, and considering the coarse-granular overlay fault diagnosis feature, we simply use original classful network ID as *NetID* of a given component. We believe the granularity given by *NetID* is sufficient for overlay applications to take necessary countermeasures. Thus, in *EUDiag* framework, each component is assigned a 136-bit unique component ID.

This reduction can significantly improve the efficiency of fault diagnosis in $EUDiag$ without losing necessary granularity. Each logic component (c_i) may consist of multiple physical hops, which may have different overall prior fault effect. We define a parameter called Component Weight (W_{c_i}), a number of physical hops contained in c_i, to describe such characteristic. For overlay nodes existing on end-user networks (EUN), their weights are all set to 1.

EUDiag does not assume knowing or be able to discover complete network members (e.g. underlying routers in ASes). For the unidentifiable network portion (e.g. routers don't response to traceroute), we compute hash function on the following 4-tuple: the ASN and NetID of the hops before and after the unidentifiable network portion.

Definition 2: Overlay Symptoms (S): Network symptoms can be classified into various categories and represented differently. In *EUDiag*, *Overlay Symptoms* are defined as end-to-end observations of network disorders (e.g. reachability outage or high packet loss ratio, abnormal latency) on overlay links. For simplicity without losing generality, we assume the category of all overlay symptoms are same (e.g. packet loss). Thus, each overlay symptom (s) can be simply represented by a sequence of overlay network components.

The symptoms could be positive or negative. Negative symptoms (also denoted as S^N), $s_i, (s_i \in S)$, indicate certain network disorder that involves at least two of the identifiable components (overlay or underlay). Negative symptoms can be generated as a result of application notification or monitoring probing. However, positive symptoms (also denoted as S^P) $\overline{s_i^o}$, which could be inferred only as a result of infrastructure monitoring probings, indicate healthy status for all identifiable components in the diagnosed path.

Definition 3: Overlay Faults (F): F is a set of faulty components. If the corresponding faulty components are overlay components, we call them overlay faults denoted as F^o. Similarly, if the corresponding components are underlay components, we call them underlay faults denoted as F^u. $F = F^o \cup F^u$. Distinguishing faults between overlay and underlay components has significant impact on overlay fault recovery. Generally, if the fault is caused by F^o, an overlay application may easily choose a backup overlay node from the same site. On the other hand, if the fault is caused by F^u, then an overlay application need find another overlay node [8] to bypass the faulty underlay components.

III. OVERLAY FAULT DIAGNOSIS USING BELIEF REVISION

A. User Beliefs

Overlay fault diagnosis has to be conducted from user-level. Many intelligent tools/utilities were proposed for end users to generate more informative symptoms [32][31][27]. However, how to systematically combine/integrate *User Beliefs* to find best explanation for observed symptoms is an unresolved issue.

Belief Function can be defined specifically based on the given problem domain. To make it more applicable, *EUDiag* is designed based on four most common but critical *User Beliefs* in overlay fault diagnosis domain. In the following, we first discuss these four *User Beliefs* (knowledge), then introduce a novel mechanism to compute *User Belief* value:

- *Belief-1:* Generally, overlay component has higher prior fault probability than underlay component [4][13]. Thus we have $p(1c_i) \gg p(0c_j)$. Here, $1c_i$ and $0c_j$ represent any overlay and underlay component respectively. For instance, the prior fault probability of network router can be safely estimated less than 10^{-4}. However, the prior fault probability of overlay node can be commonly greater than 10^{-2} [4]. Thus, we prefer using overlay components to explain observed overlay symptoms.
- *Belief-2:* The probability of simultaneous underlay faults is low [14][17]. The underlying faults for a logic underlay component c_i may contain multiple physical components. Thus, the prior fault probability of a logic underlay component c_i increases with the more contained physical components ($p(c_i) = 1-(1-p)^N$. p is estimated average prior fault probability of each physical component). Thus, we prefer using least weight components to explain maximum number of symptoms.
- *Belief-3:* The probability of multiple *appAgents* generate false alarms simultaneously is low [17][16]. Thus, the same component the more symptoms associated with or a hypothesis (a set of components which explains observed symptoms), the more likely the component is faulty or the hypothesis is true.
- *Belief-4:* If negative overlay symptom is received, we believe at least one component (either overlay or underlay component) is faulty; if positive symptom received, we believe all relevant components are in good status [14][16].

User-level overlay fault diagnosis is a process of reasoning root causes that best explain observed overlay symptoms by combining user beliefs. We show in the following example (as shown in Fig. 1) that the above four *User Beliefs* can cover all different user observations and provide fundamental support for user-level fault reasoning. In an extreme scenario, when overlay monitoring agent m_k only received one overlay

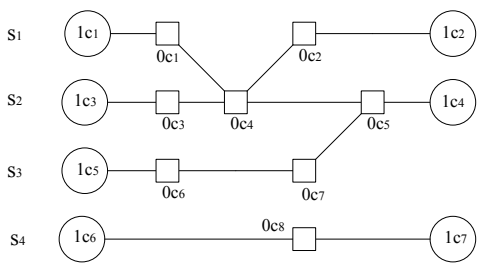

Fig. 1. User Belief Based Overlay Fault Diagnosis

symptom (e.g. s_4 in Fig. 1), we still can make reasonable fault inference that at lease one of the components (e.g. $1c_6, 0c_8, 1c_7$) is faulty based on *Belief-4*. Thus, we have the following hypotheses: $h_1 = \{1c_6\}$; $h_2 = \{0c_8\}$; $h_3 = \{1c_7\}$; $h_4 = \{1c_6, 0c_7\}$; ...; $h_7 = \{1c_6, 0c_8, 1c_7\}$. According to *Belief-2*, we believe h_1, h_2, h_3 are more likely to happen. Further, according to *Belief-1*, we want to conclude that h_1 or h_3 should be considered first. If we have the knowledge on prior fault probabilities ($p(1c_6$ and $p(1c_7)$) of overlay nodes $1c_6$ and $1c_7$, then one of the candidates with higher prior fault probability should be investigated first. In another example, if the monitoring agent m_k received both s_2 and s_3, according to Belief-2, we make a hypothesis $h = \{0c_3, 1c_4\}$. Then we need combining Belief-1 and Belief-2 to choose between $0c_3$ and $1c_4$. The challenge here is how to automate such intelligent analysis process by properly combining all user beliefs with appropriate and adjustable belief weights, which decide the effect of different beliefs in final decision.

B. Incremental Belief Revision

We propose a novel user-level *Belief Function* (BF) (as shown in Eq.1) that seamlessly integrates and quantify all above *User Beliefs* with the consideration of weight adjustability and belief incremental revision capability. Different overlay fault diagnosis system can adjust several system parameters to optimize the belief weights and improve the credibility of *Belief Revision* function. In Fault Reasoning module, we use the *Belief Function* (BF) as a criteria to find faulty component that have maximal indications explaining observed overlay symptoms.

$$BF(c_i) = [1 - \prod_{a \in A_{S_{c_i}}} p(a)] \times [\gamma |S_{c_i}| + (1 + PF(c_i) \times K) W(c_i)] \quad (1)$$

The *Belief Function* $BF(c_i)$ is elaborated as the following:

- S_{c_i} is the set of observed overlay symptoms that contain the same component c_i. $|S_{c_i}|$, the size of such symptom set, shows the strength of observed indications regarding if the component c_i could be faulty. For example, in Fig.1, if we received all symptoms, then $|S_{0c_4}| = 2$;
- $A_{S_{c_i}}$ is the set of different application monitoring agents ($appAgent_{ij}^k$) reporting the symptoms contained in S_{c_i};
- $p(a)$ is the probability that a monitoring agent ($appAgent_{ij}^k$) sends false alarms or spurious symptoms, which can be observed based the corresponding agent's historic records. For example, the ratio of the total number of reported false alarms over the total number of reported symptoms. Initially $p(a)$ is set to 0, which means the system initially trusts all monitoring agents. Thus, $(1 - \prod_{a \in A_{S_{c_i}}} p(a))$ shows that the probability that at least one monitoring agent in $A_{S_{c_i}}$ reports the true observations;
- $W(c_i)$ is the weight of component c_i, the number of individual network hops/routers in this component. The higher the value of $W(c_i)$ is, the more chance that c_i may become faulty by considering $[1 - (1 - p)^{W(c_i)}]$ as the probability of one physical machines contained in c_i faulty (p is the prior fault probability of a physical machine which we don't need to know).
- *PF* is an utility function to get the prefix of the component c_i to distinguish overlay and underlay components.
- K ($K \geq 1$) is the overlay weight parameter used to reflect the fact that the average overlay component prior fault probability is much higher than underlay component's. However, K should be adjusted to reflect the real system. K can be estimated by performance metrics (e.g. average upTime, average packet drop ratio);
- γ ($\gamma > 0$) is a system parameter to adjust the effect (belief weight) of estimated prior knowledge (K and $W(c_i)$) and posterior observations (S_{c_i}).

Before performing *Belief Revision* calculation, the pre-processing should be conducted to remove all components contained by S^P from the component set covered by S^N. This pre-processing could effectively reduce irrelevant components from investigated component set.

Incremental Belief Revision with New Evidence: Since individual monitoring agent may continuously receive new symptoms, and collaboration among monitoring agents may also need correlate distributed *User Beliefs*, it is important that *Belief Revision* can be incrementally conducted based on previous *User Belief* result and new evidence. Here we do not consider temporal correlations among observed symptoms and assume *Belief Revision* are conducted within valid observation time window. We mark the consideration of integrating temporal factor into *Belief Revision* as our future work.

Let $T_n = \prod_{a \in A_{S_{c_i}}} p(a)$ and $S'_{c_i} = \{s'\} \cap S_{c_i}$ (s' is the new observed symptom), we have the following incremental *Belief Revision* formula as shown in Eq.2. By using the previous *User Belief* result BF^n and T_n, as well as the spurious symptom rate of the *appAgent* that reported the new symptom, we can efficiently revise the previous *User Belief* to produce a new revised *User Belief* BF^{n+1}. This is a critical feature in achieving incremental local *Belief Revision* and global belief

aggregation and revision capabilities.

$$\begin{aligned} BF^{n+1}(c_i|S'_{c_i}) &= (1 + p(a_{s'}) + \tfrac{p(a_{s'})-T_n}{1-T_n})BF^n(c_i|S_{c_i}) \\ &\quad + (1+\gamma)T_n + (\gamma T_n - 1)p(a_{s'}) \\ &\simeq (1 + 2p(a_{s'}))BF^n(c_i|S_{c_i}) \\ &\quad + (1+\gamma)T_n + (\gamma T_n - 1)p(a_{s'}) \end{aligned} \tag{2}$$

C. Distributed & Collaborative Overlay Fault Reasoning

Once having *User Belief* on each candidate component, we need properly define a Belief Threshold B_{TH} to evaluate the obtained User Beliefs. B_{TH} is model-dependent and should be considered as a trade-off among fault detection rate, false alarm rate and detection time. For all components (c_i) with $BF(c_i) > B_{TH}$, we can dynamically create the following *Overlay Reasoning Belief Graph* (ORBG):

- For every selected overlay or underlay components with enough *User Belief* (e.g. $BF(c_i) > B_{TH}$), associate a vertex c_i;
- For every overlay symptom containing at least one component with enough *User Belief*, associate a vertex s_j
- For every selected component c_i and its related symptoms, associate an edge e_{ij} with weight equal to $BF(c_i)$;

Given ORRG, the task of fault reasoning is to find a minimal set of components (the hypotheses) that can explain all observed symptoms. This a typical Set-covering problem that has been proven a NP-hard problem. We use a greedy search approximation algorithm to fine one, or multiple hypotheses in the case all hypotheses having same *User Belief* value. Among multiple hypotheses, we choose the one with maximum value of $\sum_{c_i \in h} BF(c_i)/|h|$.

D. Credibility Evaluation of Fault Hypotheses

The fault hypotheses created by Overlay Fault Reasoning module may not accurately determine the root faults because of incomplete symptom observation. *Credibility Evaluation* is to measure the hypothesis credibility created in *Fault Reasoning* phase. How to objectively evaluate the reasoning result is crucial and also challenging. We developed a *Credibility Function* $CF(h)$ to measure the credibility of hypothesis h that used to explain observed symptom S.

$$CF(h) = \frac{\sum_{c_i \in h} BF(c_i)}{\sum_{s_i \in S}(2K + \sum_{c_i \in C_{s_i}} W(c_i))} \tag{3}$$

$CF(h)$ is essentially the likelihood measurement of the hypothesis (h) by averaging total *User Beliefs* over all components contained in observed symptoms S. Here, K is the overlay weight factor, and each overlay symptom involves two overlay nodes. Since the maximal credibility value can not be obtained because of the incompleteness property of *ORBG*, intuitively, the more relevant symptoms observed and less relevant components involved, the higher credibility can be achieved. We develop a credibility algorithm taking into consideration a target *Credibility Threshold*, $C_{threshold}$, that the user can configure to accept hypothesis. The initial observation is usually not sufficient. Active actions have to be conducted to enhance or reduce the belief on the proposed hypothesis. If the threshold is set too high, even correct hypothesis will be ignored; but if the threshold is too low, then less credible hypothesis might be selected. Overlay network administrators can define the threshold based on long-term observation and previous experience (e.g. detection rate and false alarm ratio).

E. Action Selection Heuristic Algorithm

We take active actions whenever initial symptom observation is not sufficient or reasoning result is not satisfactory. The action results could provide more relevant symptoms (either positive or negative) which can be used to (1) collect more evidence to increase *User Belief*; or (2) verify the correctness of given hypothesis. Active actions could be as common as common network utilities Ping, traceroute, or specifically designed network tools [32], [27], [31], [30]. Different overlay application may offer different action set based on various factors, such as running platform, availability. Each action can be associated with a cost value (denoted as T_i) administratively with the consideration such as network intrusiveness or coverage (e.g. number of hops), overlay node importance, security risk, etc..

For simplicity, we use C to represent components need be verified in the following. Given a set of components $C = \{C_1, C_2 \cdots, C_N\}$, and a set of actions $A = \{A_1, A_2, \cdots, A_M\}$, here each action has its coverage denoted as: $A_i = \{C_x, C_y, \cdots, C_z\}$ ($A_i \in A$). Obviously $C \subseteq \bigcup_{A_i \in A} A_i$. Action Selection is to find a set of actions A' ($A' \subseteq A$) such that: (1) all components in C are covered; (2) the total cost $T = \sum_{A_i \in A'} T(A_i)$ is minimized.

The task of Action Selection is to find the least-cost actions to verify all components C included in the hypothesis that has highest credibility. As the size of C could grows very large, the process of selecting the minimal cost action that verifies C becomes non-trivial. This problem can be modelled as a weighted set-covering problem, which is NP-complete. Thus, we developed a heuristic greedy set-covering approximation algorithm to solve this problem.

IV. SYSTEM EVALUATION

In this section, we present our evaluation metrics, simulation methodology and simulation results.

A. Evaluation Metrics

The performance and accuracy are the two most important factors for evaluating fault diagnosis techniques. Performance is measured by fault detection time T, which is the time between receiving the fault symptoms and identifying the root faults. The fault diagnostic accuracy depends on two factors: (1) the detection ratio (α), which is the ratio of the number of *true* detected root faults (F_d is the total detected fault set) to the number of *actual* monitored and occurred faults F_h, formally $\alpha = \frac{|F_d \cap F_h|}{|F_h|}$; and (2) false positive ratio (β), which is the ratio of the number of *false* reported faults to the total number of detected faults; formally $\beta = \frac{|F_d - F_d \cap F_h|}{|F_d|}$ [17]. The

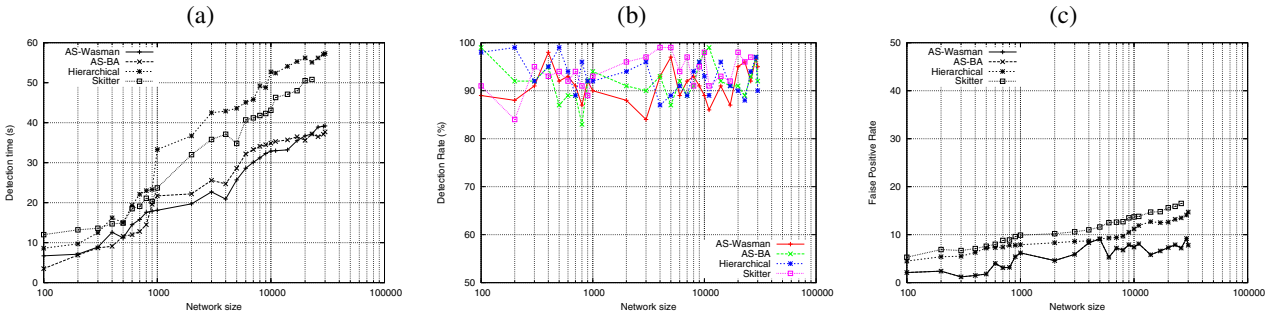

Fig. 2. **The Impact of Network Topology and Size (a) Detection time T (b) Detection rate α (c) False positive rate β**

scalability and robustness are also critical for a distributed fault diagnosis system.

B. Simulation Methodology

We consider the following dimensions and parameters for simulation.

- *Underlying Network Topology and Size:* We use a synthetic topology generator BRITE [23] with three types of topology model. The number of nodes ranges from 100 to 30,000. In addition, we import to BRITE using real network AS topology data from Skitter [24] with each set more than 20,000 ASes for evaluation.
- *Overlay Network Topology:* The distribution of overlay nodes provides different observation points and may significantly impact the performance of *EUDiag*. We define three different overlay topologies: edge overlay (overlay nodes located on edge networks); core overlay(overlay nodes located on core networks); mixed overlay (overlay nodes located on both edge and core networks).
- *Overlay Weight K and system parameter γ:* K and γ are model-dependent and should be properly adjusted to reflect the real system characteristics. We assign overlay nodes with prior fault probability uniformly distributed among 1-30%.
- *Link Loss Distribution:* We adopt $LLRD_1$ model as shown in [11], 95% links are classified as reliable with prior fault probability uniformly distributed among $0-0.3\%$; 5% links are classified as faulty with prior fault distributed on $5-10\%$.

C. The Impact of Network Topology and Size

There are four types topologies simulated by using BRITE [23]: (1) AS-level Waxman model; (2) AS-level Barabasi-Albert model; (3) Hierarchical model; (4) Skitter [24]. For the first three topology model, we simulate three different scenarios: (1) use small-size network (nodes between 100 and 1,000 and placed in relatively small area of the plane) to simulate regional collaborative ISPs and their overlay networks; (2) use medium-size network (nodes between 1,000 and 10,000 and wide-distributively placed in the plane) to simulate national collaborative ISPs and their overlay networks; (3) use large-scale network (nodes between 10,000 and 30,000) to simulate the Internet. For each generated topology, we select 10% of total nodes as overlay nodes with the three distribution: edge, core and mixed overlay topology. Then we run simulation 10 times independently. We found the different overlay topologies showing similar results with given underlay topology. In the following, we will only show the results using mixed overlay topology.

The simulation results are shown in Fig.2. Apparently with the increase of network size, the more chance that the network components have problems. With the properly integration of active actions and passive analysis, when the network size increased 1000% from small-sized network to medium-sized network; and further from medium-sized network to large-scale network, the corresponding detection time is just increased only 2 times (approximately from 10 to 20 seconds) and 3 times respectively (approximately from 20 to 60 seconds). For the detection rate and false positive rate, the change rate is within 10% with the increase of network size. As shown in Fig.2, there are no evident difference for different network topology.

V. INTERNET EXPERIMENTS

The motivation of this work is to tackle new challenges in overlay fault diagnosis. We are interested (1) to show the performance of *EUDiag* in the real system; (2) to prove collaboration with information sharing among overlay applications and overlay service providers an effective approach in overlay fault diagnosis.

TABLE II
EXPERIMENTAL OVERLAY TOPOLOGY

Planet-Lab Node Dist	OSP-1			OSP-2		
	1	2	3	1	2	3
N. America (edu) (346)	30	55	25	30	0	25
N. America (non-edu) (28)	3	10	4	3	0	4
S. America (18)	4	0	2	4	5	2
Europe (230)	20	0	20	20	35	20
Asia (138)	7	0	6	7	25	6
Oceania (9)	2	0	2	2	4	2

A. Methodology

We implemented *EUDiag* on the Planet-Lab. Based on geographic locations, Planet-Lab nodes are mainly distributed into 5 zones: North America, Source America, Europe, Asia

and Oceania zones. North America nodes can be further classified as EDU and non-EDU nodes. Thus, all Planet-Lab nodes can be classified into 6 categories as shown in TABLE II. We created two OSPs (denoted as OSP-1 and OSP-2) on Planet-Lab testbed and generated 3 test scenarios:

- Scenario 1: OSP-1 and OSP-2 randomly choose various number of nodes from each category respectively as shown in TABLE II.
- Scenario 2: OSP-1 only chooses North America nodes; however, OSP-2 chooses non North America nodes.
- Scenario 3: OSP-1 first chooses nodes from each category; then OSP-2 chooses nodes belonging also from each category but in different institutes, companies or countries from those nodes already selected by OSP-1.

In each scenario, for each OSP-i (i = 1, 2), we created three overlay applications (denoted as App-i1, App-i2 and App-i3) and each application selects 12 overlay application nodes. For each overlay application App-ij (j = 1, 2, 3), it adopted two different topologies: well-distributed and random topology. For well-distributed topology, we evenly chose application nodes from each zone; for random topology, we randomly chose application nodes from all available overlay nodes. In all scenarios, each application randomly chooses one node per geographic category if possible as its application agent. Among all application agents, we randomly choose one as the monitoring agent. The information about selected application agents and monitoring agents is sent to all relevant nodes to report observed symptoms.

For each application, first it measured the topology among application nodes by simultaneously running "traceroute" and converted discovered networks (including unidentifiable network portions) to overlay component IDs. Each node sends the information about the discovered network (i.e. a serial of component IDs) to all destination nodes. There are $12 \times 11 = 132$ overlay links for each overlay application. For selected links, the source nodes will send UDP packets to the destination nodes. For each experiment, we measured 100 trials and each trial lasts 15 seconds. During a trial, each selected node sends 100 40-byte UDP packets with sequence number and sending time.

B. Experiment Results

1) Observation Incompleteness Penalty: For each overlay application, we control the Observation Ratio (OR) (the ratio of monitored overlay links to the total links in a given application). We change OR for each application increased from 10% to 100% with the increase rate 10%. In this way, the overall Observation Ratio of whole OSP is also controlled accordingly. We found that when OR is relatively low (10-30%), *EUDiag* required significantly increased actions (intrusiveness) (320-470%) to detect faulty components.

2) Collaboration Gain: In our experiment scenario 1, we created two case studies: (1) OSP-1 and OSP-2 running *EUDiag* independently; (2) OSP-1 and OSP-2 collaboratively running *EUDiag*. We repeated the same experiment 10 rounds and 10 times in each round for scenario 2 and 3. From Fig. 3.

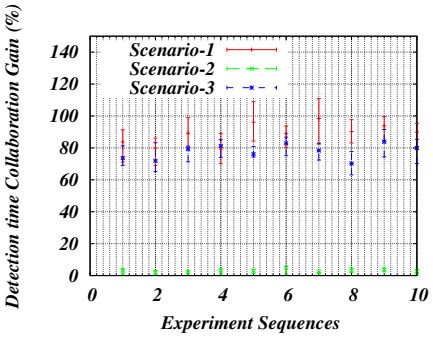

Fig. 3. Collaboration Gain on Detection Time

The collaboration gain in Scenario 1 and 3 are much higher than scenario 2. From the node distribution in scenario 2, the two OSPs are not sharing underlying infrastructure. From another perspective, it shows that distributed approach is more appropriate for largely dispersed nodes.

3) Posterior and Prior Factors in User Belief Revision: In Eq. 1, Posterior and Prior User Belief can be combined seamlessly and adjustable by two parameters: K and γ. Based on the general monitoring statistics presented in [4], overlay nodes are getting more and more reliable but still experiencing high fault probability. In our experiment in the Planet-Lab, K has been selected between 5-20 and γ is chosen between 1-3, which can make *EUDiag* function to produce stable results.

VI. RELATED WORK

In our related work study, we focus on user-level network measurement and fault diagnosis tools/approaches particularly for overlay networks. In this section, we classify user-level fault diagnosis related work into the following categories:

Passive Approach: Various passive monitoring and event correlation models were proposed including rule-based analyzing system [21], model-based system, case-based diagnosing system and model traversing techniques. In [19], a model-based event correlation engine is designed for multi-layer fault diagnosis. In [15], coding approach is applied to deterministic model to reduce the reasoning time and improve system resilience. An interesting incremental event-driven fault reasoning technique is presented in [16] and [17] to improve the robustness of fault localization system.

Measurements Diagnosis Tools: Many end-to-end traffic measurement tools were proposed for monitoring packet loss and other path properties for problem diagnosis such as [30], [32], [31]. These tools are good for diagnosis a specific network property and not adequate as a general problem diagnosis in overlay networks. Recently, some researchers incorporate active probings into fault localization. In [18], an active probing fault localization system is introduced, in which pre-planned active probes are associated with system status by a dependency matrix. An on-line action selection algorithm is studied in [18] to optimize action selection. Most of these techniques causes an extensive intrusiveness due to active

probing and at the same time may not discover intermittent problems.

Diagnosis Framework: One of the most recent interesting work is the Tomography-based approach that estimates network performance parameters based on traffic measurement at a limited subset of the nodes [28], [26]. However, similar to previous tools, this is still purely active approach which usually requires extensive probing in order to achieve accurate results regardless of problem exists or not. Anther one is the Multiple Vantage Point Approach or PlanetSeer [29] that locates Internet faults by selectively and periodically invoking "traceroute" from multiple vantage points. The measurement model is manually managed and only matches the application domain direction of data flow. Unlike these approaches which use active monitoring constantly to discover problems when occurs, our approach exploits the correlation of naturally observed symptoms to identify problem or necessary actions are initiated for fine-grain diagnosis.

To the best of our knowledge, EUDiag is the first user-level overlay fault diagnosis framework that integrates active monitoring with passive fault reasoning based on dynamic revising *User Beliefs*.

VII. CONCLUSION AND FUTURE WORK

In this paper, a novel *User Belief* based approach (called *EUDiag*) that could dynamically and incrementally encode user common belief with investigated overlay and underlay components, and it also seamlessly integrates passive and active fault reasoning in order to reduce fault detection time as well as improve the accuracy of fault diagnosis. In our future work, we will study symptom observation temporal factor and the inherent correlations between different types of symptoms and investigate how integrated symptom analysis and fault reasoning can improve the performance as well as the accuracy of overlay fault localization.

ACKNOWLEDGMENTS

This work has been partially supported by the National Basic Research program of China 973 under Grant No. 2009CB320505, the Natural Science Fundamental Program of Jiangsu Province under Grant No. BK2008288, the Excellent Youth Teacher of Southeast University Program under Grant No. 4009001018.

REFERENCES

[1] LARRY PETERSON, TOM ANDERSON, DAVID CULLER, AND TIMOTHY ROSCOE. A Blueprint for Introducing Disruptive Technology into the Internet, *In the Proceedings of ACM HotNets-I Workshop*, (July 2002).
[2] Akamai *Technology Overview*, http://www.akamai.com/en/html/technology.
[3] PlanetLab: http://www.planet-lab.org.
[4] CoMon: http://comon.cs.princeton.edu/.
[5] Level 3: http://www.level3.com/.
[6] LIMIN WANG, KYOUNGSOO PARK, RUOMING PANG, VIVEK S. PAI, LARRY PETERSON. Reliability and Security in the CoDeeN Content Distribution Network, *Proceedings of the USENIX 2004 Annual Technical Conference*, (June 2004).
[7] MICHAEL J. FREEDMAN, ERIC FREUDENTHAL, AND DAVID MAZIRES. Democratizing Content Publication with Coral, *In Proc. 1st USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI '04)*, (March 2004).
[8] DAVID G. ANDERSEN, HARI BALAKRISHNAN, M. FRANS KAASHOEK, ROBERT MORRIS. Resilient Overlay Networks, *Proceedings of the 18th ACM SOSP*, (Banff, Canada, October 2001).
[9] YONGNING TANG, EHAB S. AL-SHAER, RAOUF BOUTABA. Active Integrated Fault Localization in Communication Networks, *IEEE/IFIP Integrated Management (IM'2005)*, (May 2005).
[10] YONGNING TANG, EHAB S. AL-SHAER, BIN ZHANG. Toward Globally Optimal Event Monitoring & Aggregation For Large-scale Overlay Networks, *IEEE/IFIP Integrated Management (IM'2007)*, (May 2007).
[11] VENKATA N. PADMANABHAN, LILI QIU, HELEN J. WANG. Server-based Inference of Internet Link Lossiness, *In Proc. of IEEE INFOCOM*, (New York, NY, 2003)
[12] GEORGE J. LEE, LINDSEY POOLE. Diagnosis of TCP Overlay Connection Failures using Bayesian Networks *SIGCOMM06 Workshops September, 2006, Pisa, Italy*
[13] VERN PAXSON. End-to-End Routing Behavior in the Internet, *IIEEE/ACM Transactions on Networking*, (1996).
[14] M. STEINDER AND A. S. SETHI . A Survey of Fault Localization Techniques in Computer Networks, *Science of Computer Programming, Special Edition on Topics in System Administration*, (Vol. 53, 2 Nov., 2004)
[15] S. KLIGER, S. YEMINI, Y. YEMINI, D. OHSIE, AND S. STOLFO. A coding approach to event correlation, *Proceedings of the Fourth International Symposium on Intelligent Network Management*, (1995).
[16] 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)
[17] M. STEINDER AND A. S. SETHI . Probabilistic Fault Diagnosis in Communication Systems Through Incremental Hypothesis Updating, *Computer Networks Vol. 45, 4 pp. 537-562*, (July 2004)
[18] I. RISH, M. BRODIE, N. ODINTSOVA, S. MA, G. GRABARNIK. Real-time Problem Determination in Distributed Systems using Active Probing, *IEEE/IFIP (NOMS)*, (Soul, Korea, 2004).
[19] K. APPLETY et al. Yemanja - a layered event correlation system for multi-domain computing utilities, *Journal of Network and Systems Management*, (2002).
[20] Z. M. MAO ANDet al. Scalable and accurate identification of as-level forwarding paths, *in IEEE Infocom*, (2004).
[21] G. LIU, A. K. MOK, AND E. J. YANG. Composite events for network event correlation, *Integrated Network Management VI, pages 247260*, (Boston, MA, May 1999).
[22] YANG-HUA CHU, SANJAY G. RAO, AND HUI ZHANG. A Case for End System Multicast, *Proceedings of ACM SIGMETRICS,*, (Santa Clara, CA, June 2000).
[23] A. MEDINA, I. MATTA, AND J. BYERS. On the origin of power laws in Internet topologies, *ACM Computer Communication Review*, (Apr. 2000).
[24] Skitter, CAIDAs topology measurement tool, *http://www.caida.org/tools/measurement/skitter/.*
[25] FRED HOWELL AND ROSS MCNAB. simjava: a discrete event simulation package for Java with applications in computer systems modelling, *First International Conference on Web-based Modelling and Simulation*, (Jan. 1998).
[26] YAO ZHAO, YAN CHEN, AND DAVID BINDEL. Towards Unbiased End-to-End Network Diagnosis, *in Proc. of ACM SIGCOMM*, (2006).
[27] YAO ZHAO AND YAN CHEN. A Suite of Schemes for User-level Network Diagnosis without Infrastructure, *in Proc. of IEEE INFOCOM*, (2007).
[28] M. COATES, A. HERO, R. NOWAK, AND B. YU. Internet Tomography, *IEEE Signal Processing Magazine, vol. 19, no.3*, (2002).
[29] M. ZHANG, C. ZHANG, V. PAI, L. PETERSON, AND R. WANG. PlanetSeer: Internet Path Failure Monitoring and Characterization in Wide-Area Services, *roc. Sixth Symposium on Operating Systems Design and Implementation*, (December 2004).
[30] STEFAN SAVAGE. Sting: a TCP-based Network Measurement Tool, *Proceedings of the 1999 USENIX Symposium on Internet Technologies and Systems*, (October 1999).
[31] K. G. ANAGNOSTAKIS, M. B. GREENWALD, AND R. S. RYGER. cing: Measuring network-internal delays using only existing infrastructure, *IEEE INFOCOM*, (2003).
[32] R. MAHAJAN, N. SPRING, D. WETHERALL, AND T. ANDERSON. User-level Internet path diagnosis, *in Proc. ACM SOSP*, (October 2003).