Characterizing VLAN-Induced Sharing in a Campus Network

Mukarram Bin Tariq, Ahmed Mansy, Nick Feamster, Mostafa Ammar
{mtariq,amansy,feamster,ammar}@cc.gatech.edu
School of Computer Science, Georgia Tech. Atlanta, GA

ABSTRACT

Many enterprise, campus, and data-center networks have complex layer-2 virtual LANs ("VLANs") below the IP layer. The interaction between layer-2 and IP topologies in these VLANs introduces hidden dependencies between IP level network and the physical infrastructure that has implications for network management tasks such as planning for capacity or reliability, and for fault diagnosis. This paper characterizes the extent and effect of these dependencies in a large campus network. We first present the design and implementation of EtherTrace, a tool that we make publicly available, which infers the layer-2 topology using data passively collected from Ethernet switches. Using this tool, we infer the layer-2 topology for a large campus network and compare it with the IP topology. We find that almost 70% of layer-2 edges are shared by 10 or more IP edges, and a single layer-2 edge may be shared by as many as 34 different IP edges. This sharing of layer-2 edges and switches among IP paths commonly results from trunking multiple VLANs to the same access router, or from colocation of academic departments that share layer-2 infrastructure, but have logically separate IP subnet and routers. We examine how this sharing affects the accuracy and specificity of fault diagnosis. For example, applying network tomography to the IP topology to diagnose failures caused by layer-2 devices results in only 54% accuracy, compared to 100% accuracy when our tomography algorithm takes input across layers.

Categories and Subject Descriptors: C.2.3 [Computer Communication Networks]: Network Operations, Network Management

General Terms: Management, Measurement, Reliability

Keywords: Network Diagnosis, Network Virtualization, VLAN, VLAN-induced dependency

1. INTRODUCTION

Virtual LANs (VLANs) enable many distinct LANs to coexist on a fixed set of physical switches and links. Enterprise, campus, and data-center networks use VLANs to group hosts into common administrative or functional units, independent of their location in the physical network topology. For example, a campus network configuration may place all students on a common VLAN to make it easier for a network administrator to apply common policies to the group of users. VLANs offer network operators flexibility for specifying management and security policies within an enterprise and allow operators to implement some level of isolation by separating hosts into different broadcast domains. This flexibility, however, comes at some cost: IP layer paths that are otherwise disjoint may be "trunked" at layer 2, thereby introducing sharing between paths that might have otherwise experienced independent performance and failure characteristics (e.g., if they had not shared physical infrastructure). Moreover, because these layer-2 paths are not visible at the IP layer, this sharing may make it more difficult to diagnose some performance problems with conventional IP-layer tools (e.g., traceroute).

To properly design and debug their networks, network operators need tools that provide some visibility into network paths that share common network elements at lower layers (e.g., layer-2 switches or middleboxes). Unfortunately, current understanding of this sharing—how to measure and characterize it, as well as its effects on network reliability, troubleshooting, and diagnosis—is poor. If operators could instead have better visibility into cross-layer dependencies, they might be able to better design their networks to avoid dependencies that might compromise redundancy, and they might also be able to diagnose performance or availability problems more quickly by identifying common lower-layer network elements on paths that might appear independent at higher layers.

This paper does not tackle the challenging question of network design. Instead, we take an initial first step in cross-layer analysis, focusing on how VLANs create sharing and dependencies between IP-layer paths that are otherwise disjoint. We present a preliminary study that characterizes the dependencies that exist among IP subnets that run over the VLANs on a large campus network. We also characterize the causes of this sharing, as well as the implications of sharing for both reliability and network fault diagnosis. Towards this goal, this paper presents three contributions:

- ***EtherTrace*, a passive layer-2 topology discovery tool.** *EtherTrace* infers the VLAN (layer-2 network) topology using mostly passive measurements of bridge and ARP tables from the switches in the network. We have made *EtherTrace* publicly available [7].

- **An empirical analysis and characterization of VLAN-induced sharing in a large campus network.** We analyze VLAN-induced sharing across IP network segments for the Georgia Tech campus network. The topology discovery and analysis uses bridge-table entries from 1,461 switches corresponding to about 29,000 active MAC addresses spanning

1,126 unique VLANs, as well as IP traceroutes between hosts on 79 distinct subnets across the network. Our analysis of the layer-2 topology finds that as many as 85% of layer-2 edges shared by at least 6 distinct IP edges, and one layer-2 edge was shared by as many as 34 distinct IP edges. Most common reason for such sharing is trunking multiple subnets using VLAN trunking to aggregation routers. We also find instances where IP subnets do not share routers but nevertheless share layer-2 infrastructure: these cases arise due to multiple academic departments sharing a building and layer-2 infrastructure, but having separate IP subnets.

- **An analysis of the effects of sharing on tomography-based diagnosis, and a preliminary analysis of a cross-layer diagnosis algorithm.** We quantify how layer-2 dependencies can introduce inaccuracies in network tomography algorithms and demonstrate how a cross-layer approach can improve accuracy by a factor of two and specificity by a factor of four.

In Section 2, we present background on VLANs and an overview of *EtherTrace*. In Section 3, we apply *EtherTrace* to the Georgia Tech campus network. In Section 4, we analyze the extent and nature of the shared dependencies between IP segments and characterize the types of sharing that VLANs can induce. In Section 5, we describe the cross-layer fault localization method to improve the accuracy and specificity of the inference. We review related work in Section 6 and conclude in Section 7.

2. LAYER-2 TOPOLOGY DISCOVERY

This section presents a brief overview of VLANs and describes *EtherTrace*, a tool for layer-2 topology inference.

2.1 VLAN Background

Ethernet-based Local Area Networks (LANs) use learning bridges to connect network segments. These bridges use a spanning-tree algorithm to achieve a loop-free topology. The bridges learn the location of hosts on the network by “listening” for the frames sent by the hosts; if a bridge hears a frame sent by host *A* on a port *x*, then the bridge forwards all the future frames for *A* to port *x*. Bridges maintain this information in bridge tables, where each entry in the table has the MAC address of the host and the port on which the frames from the host were received. Bridge tables are dynamic: table entries timeout if they are not refreshed by new frames from the hosts; we call the hosts *active* if their MAC addresses are refreshed.

On *Virtual LANs (VLANs)*, a single bridge may be simultaneously assigned to one or more VLANs. The bridges on each VLAN form separate spanning trees, thus enabling multiple virtual topologies to co-exist on the network. To support VLANs, the entries in bridge tables also include the 12-bit VLAN identifier, which is copied directly from the frames. Bridges forward traffic within each VLAN using the bridge tables but do not forward traffic across VLANs. A router connects VLANs to allow inter-VLAN communication. Most modern routers and switch-router devices support VLAN interfaces, which allows these devices to have their interfaces appear as multiple virtual interfaces, each on a separate VLAN.

A common use of VLANs in enterprises, campus, and data-centers is to group hosts in administrative domains or functional units at the IP level, irrespective of their physical location in the topology. The IP subnets are essentially overlays on the physical layer-2 infrastructure, and multiple IP subnets may share underlying layer-2 infrastructure. Although the entire layer-2 infrastructure may be owned and operated by a single network entity, maintaining the mapping between layer-2 devices and paths for IP subnets can be challenging because of size and changes in configuration over time. Unlike IP networks, where ICMP allows for topology discovery, there is no standard method for discovering layer-2 topology. Mechanisms such as Cisco Discovery Protocol (CDP) [3] can perform active traceroutes on layer-2, but they require all switches in the network to run CDP, which is seldom the case, due to the heterogeneity of devices in the network. This necessitates developing new systems that can infer layer-2 topology and its mapping to IP network using interfaces that are widely supported by layer-2 switches. In the following section, we describe *EtherTrace*, a layer-2 topology discovery tool that uses bridge-table entries obtained from switches using standardized SNMP interfaces.

2.2 EtherTrace Algorithm and Tool

EtherTrace is a passive layer-2 topology discovery algorithm that infers the layer-2 elements on an IP path by inspecting bridge-table entries in the network. *EtherTrace* relies on the following simple observation to discover layer-2 devices corresponding to each IP path segment: Because the bridges on LAN form a tree, only one layer-2 path between any pair of hosts exists at any time. Thus, the bridges along the path between two hosts on the same VLAN must always receive frames from those two hosts on two separate ports. On the other hand, bridges that are not along the path between the hosts will always receive the frames from the two hosts on the same port.

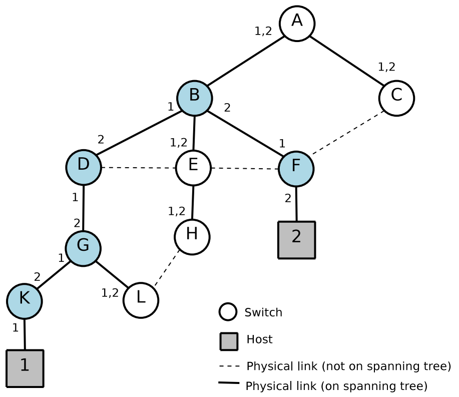
Figure 1 illustrates this observation. In this figure, the topology comprises 12 switches (labelled A–L). The physical links that are on the spanning tree are shown as solid lines, and the blocked links are shown as dotted lines. We assume that there are two hosts on this network. The switch ports are labelled with the identifier of the host whose frames are received on that port. To determine the layer-2 path between host 1 and host 2, we observe that the switches that are along the path between hosts 1 and host 2 receive MAC address of hosts 1 and host 2 on two different ports; all other switches receive the MAC addresses on the same port. *EtherTrace* can determine the set of all switches along a IP path as a union of constituent switches along all IP path segments. *EtherTrace* can also determine the order of switches and ports along a path but the order is usually not important for dependency analysis. Thus, we present only the method for determining the set of layer-2 switches and ports on a path. We formally describe the method below.

Notation. We refer to a snapshot of bridge-table entries from all the switches in the network for a particular host x as T_x. Each entry, $e \in T_x$, is a 3-tuple $(e.b, e.p, e.v)$, where the three elements refer to the bridge, the port, and the VLAN-identifier fields, respectively. For hosts x and y, $T_{x\Delta y}$ refers to the symmetric difference[1] of sets T_x and T_y. $B(T_x)$ and $V(T_x)$ refer to the set of bridges and set of VLANs that the entries in T_x refer to, respectively.

Determining the path elements. To determine the switches on the IP path between two hosts, *EtherTrace* first uses the ARP tables to obtain the MAC addresses of the two hosts. To determine the layer-2 path, *EtherTrace* considers two cases.

Case 1: Hosts on the same VLAN and IP subnet. From the bridge tables, *EtherTrace* determines the bridge-table entries involving the two hosts, and among these entries determines whether for some VLAN identifier(s), there exists a set of switches that receive the MAC addresses of the two hosts on separate ports on that VLAN.

[1]The symmetric difference of sets A and B is $A \cup B - A \cap B$.



A
B
C
D
E
F
G
H
K
L
1
2
1,2
Switch
Host
Physical link (not on spanning tree)
Physical link (on spanning tree)

Figure 1: Port labels show the hosts whose frames are received on that port. Switches that are along the path between hosts 1 and 2 (i.e., the shaded nodes) receive the frames from these hosts on two different ports. Other switches receive the frames from the hosts on the same port, if at all.

Specifically, for hosts x and y, the elements on the path are given as a set of 3-tuples as follows:

$$\hat{S}(x,y) = \{(e.b, e.p, e.v) : \; e \in T_{x\Delta y}, \\ e.b \in B(T_x) \cap B(T_y), \\ e.v \in V(T_x) \cap V(T_y)\} \qquad (1)$$

The first constraint in Equation 1 selects all of the bridge, port, VLAN tuples that belong to either hosts, except those that are identical (i.e., where the bridge hears from the two hosts on the same port and VLAN tag). The second and third constraints ensure that *EtherTrace* includes only the bridges that receive frames from both hosts on same VLANs.

There are two sub-cases. Although network administrators usually configure each VLAN to correspond to one IP subnet, it is possible to connect multiple VLANs by connecting two non-trunk ports with a loop cable, in which case the spanning-trees on the VLANs merge. If the two hosts are on such VLANs, the above algorithm will still work correctly because the merged VLANs will appear in $V(T_x) \cap V(T_y)$. The second sub-case occurs when a pair of bridges along a path are connected through a passive component, such as a hub or a repeater element. *EtherTrace* cannot recognize such passive elements. If the two hosts are on such a segment, *EtherTrace* declares an empty path between the hosts. In most modern network deployments, each host connects directly to a switch port and there is little, if any, communication that takes place over hubs or buses. As a result, *EtherTrace* will fail in only a few cases. Similar to this sub-case, it is possible that *EtherTrace* is not aware of presence of some switches in the network and thus does not obtain bridge tables from them. *EtherTrace* is unable to detect such switches if they appear on the path.

Case 2: Hosts on different VLANs. When hosts are on different VLANs on the network, *EtherTrace* can determine the layer-2 path between these hosts by using the IP-level traceroute between these hosts. *EtherTrace* first infers the layer-2 path elements for each IP-path segment and then concatenates them to determine elements on the entire path. If the traceroute between the hosts x and y is $\{h_1 \cdots h_k\}$, $h_1 = x$ and $h_k = y$ then the set of layer-2 elements on the path is:

Data Sources	
Switches	1,358 successfully polled for this dataset. (1,461 total in the network.)
Routers	31
CPR Nodes	79
Dataset	
Bridge Tables	Polling Interval: 4 hours.
ARP Tables	Polling Interval: 1 hour. 28,836 active MAC addresses; 88,932 including stale.
IP Trace-routes	Once every five minutes.

Table 1: Summary of the dataset.

Duration	$< 1s$	$< 2s$	$< 5s$	$< 20s$	$< 40s$	$< 57s$
Switches	700	986	1114	1280	1340	1358
	48%	67%	76%	87%	91%	93%

Table 2: Latency for obtaining bridge-table entries from the switches. 93% of switches replied within one minute.

$$S(x,y) = \bigcup_{i=1:k-1} \hat{S}(h_i, h_{i+1}) \qquad (2)$$

Implementation. We have implemented *EtherTrace* in Python with a MySQL database backend and made it publicly available [7]. The current implementation of *EtherTrace* relies on a database that is continually updated with the bridge tables from switches, ARP tables from routers, and traceroute information from the hosts in the network. Obtaining the bridge tables from the switches and ARP tables from the routers requires administrative access to these devices, but once the intermediate database is populated with these tables, the inference is passive and does not require interaction with the network devices. As a result users do not require administrative access to the network devices to obtain the layer-2 paths.

3. DATA AND TOPOLOGY

This section describes the data and the process of inferring the layer-2 topology using *EtherTrace*.

3.1 Data

Types of data. We rely on three sources of data, all from the Georgia Tech campus network. The first is the bridge table entries obtained from all the switches; we poll these switches every four hours using SNMP. The second is the ARP tables; we poll these routers hourly. These tables provide us with IP address to MAC address mappings. The third is the IP traceroutes between 79 *CPR nodes* [4], end hosts that are deployed in mostly distinct subnets on the campus network. These nodes perform pairwise traceroutes to each other once every five minutes.

Historically, this data has been collected for auditing purposes, which is why each data set has different polling intervals. The traceroute data from the CPR nodes is the only one designed for active measurement. This paper focuses on characterization of VLAN-induced dependencies in a stable network, so the slow update rates suffice for this study.

Completeness and consistency. Because the ARP and bridge table entries usually expire more frequently than our respective polling intervals, a single snapshot of the network state may not contain all the MAC addresses or bridge-table entries. To overcome this problem, we retain entries from the previous snapshots that are not overwritten in the current snapshot.

Unfortunately, retaining older entries increases the risk of inconsistencies, for two reasons. First, the topology might change between successive snapshots as a result of failures, reconfigurations, or hosts moving to different parts of the network. Similarly, the IP addresses might change between successive snapshot due to DHCP lease expirations. To mitigate these effects, we examine only snapshots from March 25, 2008, for the comparison presented in this paper. The network had no reported network failures or outages on this day. To account for inconsistencies caused by IP-address changes, we restrict our analysis to MAC addresses that map to only a single IP address across the snapshots on March 25, 2008. To mitigate inconsistencies arising from mobile hosts on the campus wireless network, we only consider the hosts on the wired network (the wireless network is a single large VLAN on the campus network).

Obtaining the SNMP snapshots from the switches takes time, which can introduce inconsistencies; thus, we study only the parts of the network where we were able to download the bridge tables within one minute. Table 2 summarizes the distribution of latency in obtaining and processing SNMP data. In addition to the latency involved in obtaining entries from a single switch, there is also a latency involved in polling all the switches; we were able to obtain a snapshot in about 15 minutes. We verified using traceroutes that the IP level topology was stable during the snapshot window.

Size of the dataset. The dataset contains traceroutes between 79 CPR nodes, bridge table entries from 114 switches, ARP tables from 31 routers, and 29 thousand unique MAC addresses. The analysis presented in this paper only uses the MAC addresses of the CPR nodes and the router interfaces.

3.2 Topology Inference

We first measure the IP topology using the IP traceroutes between the CPR nodes. We conclude that an IP edge exists between two IP routers if those routers appear as adjacent hops in any of the traceroutes between the CPR nodes. To de-alias IP addresses belonging to different interfaces of the same routers, we use information from the router and switch configurations as well as the reverse DNS entries of the interface IP addresses. This approach is feasible because we have access to all the routers and switches.

To infer the layer-2 topology, we use the *EtherTrace* algorithm described in Section 2. For the CPR nodes that are in the same VLAN, i.e., the ones that do not have an IP router between them, we apply Case 1 of the *EtherTrace* algorithm. Fewer than 5% of CPR node pairs that share a subnet. For the rest of the paths between the CPR node pairs, we apply Case 2. To infer the layer-2 path between such nodes, we obtain the layer-2 path corresponding to each of the IP hops along the IP path and concatenate these to obtain the overall path. We use the IP address of the router as it appears in the traceroute, not the one that we obtain after de-aliasing.

We note one caveat: When computing the layer-2 path within a subnet, *EtherTrace* requires the MAC addresses of the two layer-2 endpoints to be on the same VLAN. Unfortunately, traceroute's ICMP TTL time-exceeded messages use the IP address of the return interface of the router as the source address. As a result, if the IP paths are asymmetric, then the adjacent IP addresses that appear in a traceroute may belong to different subnets and therefore different VLANs, making the IP traceroute unsuitable for computing the layer-2 path. In this work, we assume that IP paths are symmetric and this problem does not arise. When this assumption does not hold, we can obtain the IP addresses of the forward interface on the router from router configuration. This approach is feasible because operators of enterprise networks typically have access to this information.

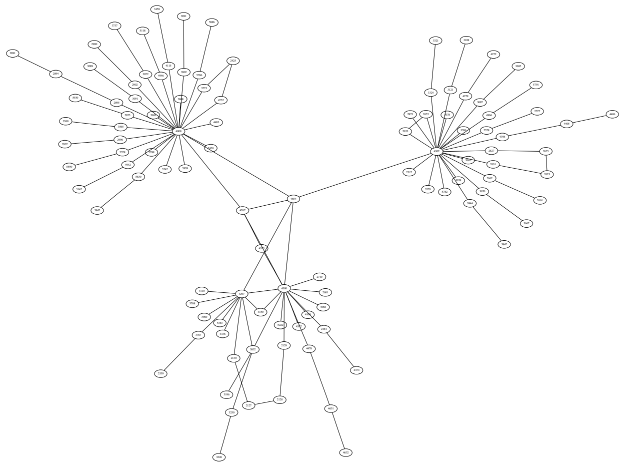


Figure 2: Layer-2 topology on the Georgia Tech network. Peripheral nodes are switches in various departments. A few core paths lead to the central routers, which provide connectivity between subnets and to the Internet.

Figure 2 shows the layer-2 topology of the Georgia Tech network as discovered using *EtherTrace*. The IP-level network structure agrees with the ground-truth network configuration that Georgia Tech's Office of Information Technology (organization responsible for network operations at Georgia Tech) provided; we have also manually verified many of these paths. The switches at the periphery belong to various departments; switches in the center lead to the core network routers that serve as the gateways between subnets and to the Internet.

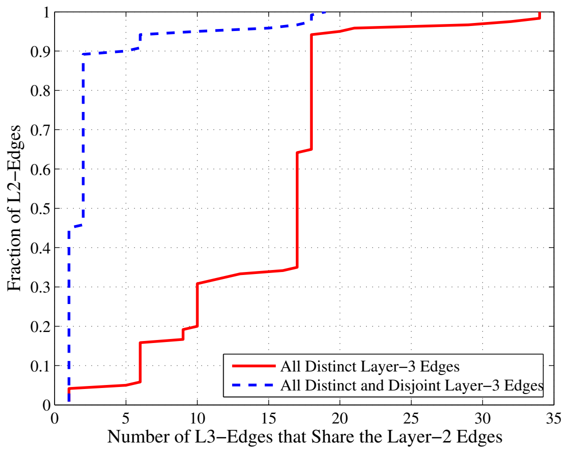
4. VLAN-INDUCED SHARING

In this section, we analyze the extent of infrastructure sharing among IP edges and paths.

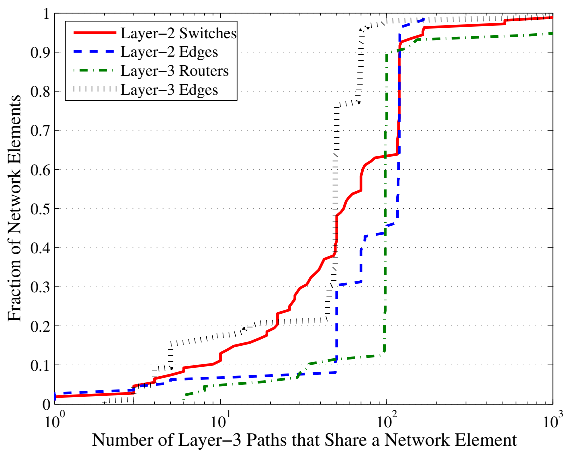
How many distinct IP *edges* traverse a given layer-2 edge? We first study sharing characteristics for all edges in the IP graph. Figure 3(a) shows the number of distinct IP edges that traverse any given layer-2 edge. The solid line in Figure 3(a) shows that 85% of layer-2 edges are shared by at least 6 distinct IP edges, and 50% of layer-2 edges are shared by 17 or more distinct IP edges. One layer-2 edge that was shared between 34 distinct IP edges. We found that this edge carried several subnets trunked to the same gateway router in the network core.

Next, we study VLAN-induced sharing for edges that do not share a router on either vertex. The dashed line in Figure 3(a) shows this distribution. About 55% of the layer-2 edges are shared by at least two IP edges that do not have a common node in the IP graph, and 4% of the layer-2 edges were shared between 17 or 18 distinct and disjoint IP edges. These situations arise when multiple subnets (VLANs) that are geographically close use a common switch to get trunked to their respective gateway routers. This setup is common when different departments co-exist in a single building (the *de facto* arrangement at Georgia Tech): each department has a logically separate gateway router, perhaps so that the departmental staff can perform independent management.

These results have significant implications for network planning



(a) Sharing of layer-2 edges among distinct IP edges.



(b) Sharing of layer-2 and IP network elements among IP paths.

Figure 3: Distribution of sharing of network elements among IP paths and edges.

for capacity and reliability. IP paths and segments that appear to be disjoint may actually not be independent because they share common underlying physical infrastructure. As a result, segments will experience correlated performance and reliability; congestion or failures in the underlying network will simultaneously affect these apparently disjoint IP paths. Indeed, determining whether two IP paths are independent often requires examining the VLAN configuration as well.

How many distinct IP-layer *paths* share a given network element? Figure 3(b) shows the distribution of the number of IP-layer paths among the CPR nodes that traverse a particular layer-2 switch, a layer-2 edge, an IP router, or an IP edge. For example, the figure shows that 50% of layer-2 switches are traversed by at least 40 IP-layer paths, and about 50% of layer-3 routers are traversed by at least 120 IP-layer paths. The less prevalent network elements exhibit less sharing of layer-2 elements than of IP elements. The more commonly occurring switches and routers have similar numbers of IP paths traversing them. In either case, the failure of a single element can affect connectivity between tens to hundreds of IP subnet pairs.

There are two causes for these characteristics. First, there are significantly more switches than routers in the network. While an IP path traverses more switches than routers, the large number of switches ultimately induces less sharing on per-element basis. Second, more frequently used routers and switches occur in the network core, and their centrality naturally induces more sharing. There is little difference in the extent of layer-2 and IP sharing among paths in the core because the switches in the core form a single IP subnet.

5. EFFECTS ON FAULT DIAGNOSIS

This section examines the implications of VLAN-induced sharing on tomography-based fault diagnosis.

Simulation. We simulate layer-2 link failures on the Georgia Tech campus topology shown in Figure 2 and use binary tomography techniques [6] to localize these failures using the set of failed IP paths as input. Given a failed edge (or edges), the method explains the failure by finding the smallest set of edges that are shared among all the failed path. The problem is analogous to finding the smallest hitting set or the set cover problem. Because finding the smallest hitting set is NP-complete, we use a randomized approximation algorithm, where we repeatedly find a random greedy solution and pick the smallest solution among the repetitions. This algorithm has an $O(log\ n)$ approximation bound, where n is the total number of edges on all the faulty paths.

The simulation proceeds as follows. For each edge on the network, we determine the IP paths between the CPR nodes that traverse the failed layer-2 edge. To measure the effects of hidden dependencies in virtualized networks, we perform fault localization using two distinct methods:

1. **Conventional approach.** As in conventional IP binary tomography, we use the IP edges on the affected IP paths that do not appear in the non-affected IP paths, as the set of dependencies for each affected IP path.
2. **Cross-layer approach.** We apply *EtherTrace* to determine the layer-2 edges corresponding to the affected IP paths that do not appear on any of non-affected IP paths as the set of dependencies for the affected IP paths.

In both cases, we use the hitting-set algorithm to find the smallest common set of edges that intersects with dependency sets of each affected IP path. We use the set of edges in the hitting set as the location of the fault. For the first method, the hitting set comprises a set of IP edges, and for the second, it comprises a set of layer-2 edges. For comparison, we convert the former to the corresponding layer-2 edges and compare the two solutions for accuracy and specificity. We compare these two approaches for every layer-2 edge in the network.

Fault localization accuracy and specificity. We define localization to be *accurate* if the hitting set that we obtain contains the original failed layer-2 edge. We define *specificity* as the multiplicative inverse of size of the hitting set; a smaller hitting set is reflected with higher specificity.

Accuracy. For the layer-2 edge failures that have a unique hitting set, the cross-layer approach yields 100% accuracy; the conventional approach yields only 54% accuracy.

Specificity. The 95th percentile of specificity using the cross-layer approach is exactly 1: the hitting set is completely specific and does not contain any extraneous edges. The average hitting set size is 1.48 layer-2 edges, or 67% specific. On the other hand, the 95th percentile of specificity using the conventional approach is only 11% (9 layer-2 edges), and the average is 27%–3.7 layer-2 edges.

Applying information about cross-layer dependencies to fault localization improved accuracy by a factor of two and specificity by a factor of four for the Georgia Tech campus network. Using only IP-level information for diagnosis results in poor specificity because the IP segments are more coarse grain than the layer-2 path segments. Using only IP-layer information reduces accuracy because an IP path segment usually has two or more layer-2 segments, and each of these segments appears as an equally likely cause.

6. RELATED WORK

Layer-2 topology discovery. *EtherTrace* relates most closely to techniques that infer layer-2 topology passively from bridge-table entries [2, 12]. The topology discovery algorithm by Lowekamp *et al.* [12] can discover a large number or Ethernet interconnections using few probing nodes that spoof their MAC addresses to inject bridge-table entries in the switches and then observe the disruptions in connectivity. Breitbart *et al.* [2] use the bridge-tables in a manner similar to *EtherTrace*, but *EtherTrace* extends this previous work because (1) it works for VLANs and (2) it uses IP traceroutes to construct topologies spanning multiple subnets. Sebos *et al.* [13] use location correlation to find shared-risk link groups of network components. In contrast, we determine the shared-risk link group based on whether IP links share the underlying layer-2 infrastructure. Cisco's Discovery Protocol (CDP) is a proprietary protocol for performing layer-2 traceroutes on a network, but it requires all switches in the network to run CDP [3].

Cross-layer diagnosis. Kompella *et al.* [10] advocate cross-layer visibility for better planning, maintenance and failure diagnosis. Several previous works have used binary [5, 6, 11] and probabilistic [1, 9] shared-risk groups for network diagnosis; this work is the first to study VLAN-induced dependencies and their implications for fault diagnosis.

VLAN characterization studies. Garimella *et al.* studied VLAN use in a campus network; they highlighted the prevalence of VLANs, characterized different types of misconfiguration, and showed that routing traffic between proximal hosts but distinct VLANs can result in significantly longer layer-3 paths [8]. In contrast, we study the relationships between IP-layer paths and layer-2 paths, and the effects of this sharing on redundancy and diagnosis. Rooney *et al.* proposed dynamic VLAN provisioning based on traffic patterns [14]; our study could inform a similar provisioning strategy for improving fault tolerance in IP networks (e.g., configuring VLANs so that disjoint IP paths are not routed over the same layer-2 infrastructure).

7. SUMMARY AND FUTURE WORK

This paper studied VLAN-induced sharing on a campus network. To study these dependencies, we designed and implemented *EtherTrace*, a passive layer-2 topology discovery tool that operates in VLAN environments; *EtherTrace* discovers the layer-2 topology for corresponding set of IP hosts. Using *EtherTrace*, we inferred the layer-2 topology for the Georgia Tech campus network, which has 94 routers, 114 switches, and almost 90,000 unique MAC addresses over the course of one day. We found that seemingly independent IP path segments can actually be dependent: some layer-2 edges were shared by more than 30 distinct IP links, and some layer-2 edges were shared by as many as 18 node-disjoint IP edges. We quantified how these dependencies can cause conventional IP-level fault localization techniques that do not consider the layer-2 topology to reach incorrect conclusions. We also showed that a cross-layer tomography approach that incorporates layer-2 and IP topology information can improve accuracy by a factor of two and specificity by a factor of four.

We believe that this initial study will motivate future work that more thoroughly examines the relationship between VLANs and IP topologies in campus and enterprise networks. For example, one area that deserves further attention is how knowledge about these dependencies could help influence network provisioning and planning: a network operator that is aware that many IP-layer paths in fact share common physical elements may decide to trunk VLANs differently or route traffic differently at the IP layer. Similarly, an operator could use knowledge of these dependencies to determine the effect of VLAN trunking decisions (e.g., sizes of bridge tables and ARP tables, traffic volumes) before those configurations are deployed. Along the lines of planning for resilience, *EtherTrace* itself could be extended to reveal nodes and links that are not part of the spanning tree but might be used in the case of failure. Designing tools—and perhaps also protocol modifications—to help operators discover the IP-to-layer-2 mappings that exist in failure scenarios could also improve network planning and operations.

Acknowledgments

We thank Russell Clark, Dan Forsyth, John Merritt, and Warren Matthews at the Office of Information Technology at Georgia Tech for their help with various aspects of this work, including making the network datasets available to us. This work was supported by NSF grants CNS-0721559, CNS-0721581, and NSF CAREER Award 0643974.

REFERENCES

[1] P. Bahl, R. Chandra, A. Greenberg, S. Kandula, D. A. Maltz, and M. Zhang. Towards highly reliable enterprise network services via inference of multi-level dependencies. In *Proc. ACM SIGCOMM*, Kyoto, Japan, Aug. 2007.

[2] Y. Breitbart, M. Garofalakis, B. Jai, C. Martin, R. Rastogi, and A. Silberschatz. Topology discovery in heterogeneous ip networks: the netinventory system. *IEEE/ACM Trans. Netw.*, 12(3):401–414, 2004.

[3] Cisco Discovery Protocol. `http://www.cisco.com/en/US/docs/ios/12_1/configfun/configuration/guide/fcd301c.html`.

[4] CPR: Campus-Wide Network Performance Monitoring and Recovery. `http://www.rnoc.gatech.edu/cpr/`, 2006.

[5] A. Dhamdhere, R. Teixeira, C. Dovrolis, and C. Diot. Netdiagnoser: Troubleshooting network unreachabilities using end-to-end probes and routing data. In *Proc. CoNEXT*, Dec. 2007.

[6] N. Duffield. Simple Network Performance tomography. In *Proc. ACM SIGCOMM Internet Measurement Conference*, Miami, FL, Oct. 2003.

[7] Ethertrace. `http://www.gtnoise.net/ethertrace/`.

[8] P. Garimella, Y. Sung, N. Zhang, and S. Rao. Characterizing VLAN usage in an Operational Network. In *Proceedings of the 2007 SIGCOMM Workshop on Internet Network Management*, pages 305–306, 2007.

[9] S. Kandula, D. Katabi, and J.-P. Vasseur. Shrink: a tool for failure diagnosis in ip networks. In *MineNet '05: Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data*, pages 173–178, New York, NY, USA, 2005. ACM.

[10] R. R. Kompella, A. Greenberg, J. Rexford, A. C. Snoeren, and J. Yates. Cross-layer visibility as a service. In *Proc. 4th ACM Workshop on Hot Topics in Networks (Hotnets-IV)*, College Park, MD, Nov. 2005.

[11] R. R. Kompella, J. Yates, A. Greenberg, and A. C. Snoeren. Ip fault localization via risk modeling. In *Proc. 2nd USENIX NSDI*, Boston, MA, May 2005.

[12] B. Lowekamp, D. R. O'Hallaron, and T. Gross. Topology discovery for large ethernet networks. In *Proc. ACM SIGCOMM*, San Diego, CA, Aug. 2001.

[13] G. H. Panagiotis Sebos, Jennifer Yates and A. Greenberg. Auto-discovery of shared risk link groups. In *Optical Fiber Comm. Conference (OFC). Volume 3. Issue 2001*, pages 1–3, 2001.

[14] S. Rooney, C. Hortnagl, and J. Krause. Automatic VLAN Creation Based on On-line Measurement. *ACM SIGCOMM Computer Communication Review*, 29(3):50–57, July 1999.